EMULATION OF THE AN/UYK-7 TACTICAL DATA COMPUTER ON THE BURROUGH'S D-MACHINE

Jerry Michael Haggerty

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

EMULATION OF THE AN/UYK-7
TACTICAL DATA COMPUTER ON THE
BURROUGH'S D-MACHINE

by

Jerry Michael Haggerty
and
John Michael Hartling

December 1976

Thesis Advisor: S. Jauregui

Approved for public release; distribution unlimited.

T176 333

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Emulation of the AN/UYK-7 Tactical Data Computer on the Burrough's D-Machine		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; December 1976 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Jerry Michael Haggerty and John Michael Hartling		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		12. REPORT DATE December 1976 13. NUMBER OF PAGES 179
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, CA 93940		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

emulation
AN/UYK-7
interpreter
microprogramming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A workable design is presented for emulating the AN/UYK-7 multiprocessing computer system on the Burrough's Interpreter Based System, the D-Machine. The program developed provides for exact execution of the AN/UYK-7 instruction repertoire with the exception of Floating Point, hardware interrupts and IOC Instructions. The design allows for future expansion to incorporate these functions. Input/Output is limited to a card

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |
(Page 1)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. (cont.)

reader, line printer and single disk. Various aspects of Emulation, the D-Machine, and the AN/UYK-7 are discussed and a detailed User's Manual is provided along with recommendations for modifying the design into a full emulation.

Emulation of the AN/UYK-7 Tactical Data Computer on the Burrough's D-Machine

by

Jerry Michael Haggerty
Lieutenant, United States Navy
B.S., United States Naval Academy, 1970

John Michael Hartling
Lieutenant, United States Navy
B.S., Miami University, Oxford, Ohio, 1969

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
DECEMBER 1976

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL

ABSTRACT

A workable design is presented for emulating the AN/UYK-7 multiprocessing computer system on the Burrough's Interpreter Based System, the D-Machine. The program developed provides for exact execution of the AN/UYK-7 instruction repertoire with the exception of Floating Point, hardware interrupts and IOC Instructions. The design allows for future expansion to incorporate these functions. Input/Output is limited to a card reader, line printer and single disk. Various aspects of Emulation, the D-Machine, and the AN/UYK-7 are discussed and a detailed User's Manual is provided along with recommendations for modifying the design into a full emulation.

CONTENTS

ACKNOWLEDGEMENTS..... 8

I. INTRODUCTION..... 9

II. EMULATION..... 12

A. GENERAL CONCEPTS OF MICROPROGRAMMING..... 13

B. BUILDING AN EMULATOR..... 17

C. APPLICATIONS..... 24

III. THE AN/UYK-7..... 27

A. INTERRUPTS..... 28

B. CENTRAL PROCESSOR..... 29

1. Program Address Register (PAR)..... 30

2. Active Status Register (ASR)..... 31

3. Base Register (S)..... 32

4. Arithmetic Register (A)..... 32

5. Index Register (B)..... 32

C. INSTRUCTION FORMAT..... 34

D. MODES OF OPERATION..... 37

IV. BURROUGH'S D-MACHINE..... 38

A. GENERAL DESCRIPTION..... 38

1. Logic Unit (LU)..... 38

2. Memory Control Unit (MCU)..... 42

3. Control Unit (CU)..... 43

4. Microprogram (M-Memory)..... 44

B. NAVAL POSTGRADUATE SCHOOL CONFIGURATION..... 47

1. Descriotion..... 47
2. I/O Interface..... 49
3. Memory Interface..... 49
C. INSTRUCTION TIMING..... 50
D. LANGUAGES..... 51
1. ALGOL..... 51
2. TRANSLANG..... 52

V. PROJECT DESCRIPTION..... 54
A. LOADER..... 56
B. EMULATION PROGRAM..... 57
C. REGISTER MAPPING..... 67
D. TIMING..... 69

VI. SUMMARY..... 71
A. PROBLEMS..... 71
B. CONCLUSIONS..... 73

VII. RECOMMENDATIONS..... 75

APPENDIX A. USER'S MANUAL..... 78

APPENDIX B. EMULATOR PROGRAM LISTING..... 91

APPENDIX C. LOADER PROGRAM LISTING.....139

APPENDIX D. SAMPLE AN/UYK-7 SORT PROGRAM.....154

APPENDIX E. SAMPLE LOADER OUTPUT LISTING.....157

APPENDIX F. SAMPLE DEBUGGER OUTPUT LISTING.....161

GLOSSARY.....167

BIBLIOGRAPHY.....175

INITIAL DISTRIBUTION.....178

ACKNOWLEDGEMENTS

We would like to take this opportunity to express our sincere appreciation to the following people for the support they have provided us over the last six months during our pursuit of this thesis. To Professor S. Jauregui, first for sponsoring the thesis, and second for providing funds for research trips to Paoli, Pa. and San Diego, Ca. without which the project could not have been done. To Mr. J. Lynch, Burrough's Advanced Development Organization (ADO), for providing software and hardware engineers on three occasions, and whose personnel graciously answered our questions during almost daily phone calls. To Mr. Carl Benson, Naval Electronics Systems Engineering Center (NESEC), San Diego, for his financial support and personal interest in the project. And last but not least our wives and families who had the patience to see us through this thesis in spite of our 14 hour days away from home, and who soothed our frayed nerves when things did not go right.

I. INTRODUCTION

In its effort to provide the Fleet with officers well versed in all aspects of Computer Science, the Naval Postgraduate School strives to furnish each student in the Computer Science Curriculum with ample opportunities for hands-on operation and programming experience on a variety of computer systems. The systems presently available for this purpose include the IBM 360/67, PDP 11/45, XDS 9300, and several varieties of microcomputers and stand alone graphic systems. Unfortunately, this wide range of available systems and their incorporated programming languages does not include any of the numerous tactical computer systems in use in the Fleet today; moreover, because of budgetary and procurement lead time constraints, it is highly unlikely that the school will have such a system in the near future. To provide the desired systems, the Chief of Naval Education and Training and the Naval Postgraduate School, with the assistance of Burrough's Corporation, have provided a medium through which students may gain experience and perform research on, not one or two Tactical Data Systems but potentially on any particular computer in which the student may express an interest. This medium is the Burrough's Interpreter Based System, a microprogrammable computer, hereafter referred to as the Burrough's D-Machine.

Through microprogramming, the D-Machine can be made to operate exactly like any designated computer of interest.

Through the process of Emulation the authors sought to demonstrate that the D-Machine could be microprogrammed to imitate a selected computer. As the target computer, the authors selected the AN/UYK-7 which is used extensively in Tactical Data Systems and is one of the more complex systems in use today. It was felt that if a realistic emulation of the AN/UYK-7 could be demonstrated, then any lower level, less sophisticated computer could also be successfully emulated.

The goal of this thesis was not a complete AN/UYK-7 emulation, but rather development of the design for the Emulation, and adoption of a sufficient subset of the AN/UYK-7 instruction repertoire to demonstrate exact and efficient execution of AN/UYK-7 programs. The design allows for a complete emulation of all AN/UYK-7 functions and provides for future expansion to a complete emulation including all IOC Functions. Chapters II, III and IV are designed to provide the reader with a general knowledge of Emulation, the AN/UYK-7 and the Burrough's D-Machine. Chapter V describes the methods used for imitating the AN/UYK-7 instructions and its various modes of operation. Chapter V also defines the mapping of the AN/UYK-7 Control Memory Registers and status bits into the Burrough's D-Machine for the purpose of this Emulation. Chapter VI presents some of the problems encountered in the course of this thesis, and

the conclusions drawn by the authors as a result of this Emulation. Chapter VII provides recommendations for expanding this Emulation and suggests possible follow on thesis topics. Appendix A is included as a User's Manual, and contains information on how to use the D-Machine, how to run AN/UYK-7 programs on the Emulator, and how to use special features of the microprogramming language TRANSLANG which were not documented in the TRANSLANG Programmer's Manual. The program listings of the Loader and Emulator, written in association with this thesis and used to demonstrate the feasibility of the design by running AN/UYK-7 Programs, are included as Appendix B and C. A sort routine written in AN/UYK-7 machine language, and used to demonstrate the Emulator's capability is provided in Appendix D. The output of the sort program and the optional Loader functions of assembler and debugger are presented as Appendix E and F.

II. EMULATION

Modern computer systems are generally composed of five basic units: input, output, memory, arithmetic/logic, and control. The instruction execution and communications among these units are usually well understood with the exception of the control unit which is, typically, understood only by the system engineer. The control unit generates the signals necessary for the information flow and timing of the system. In conventional computers this is done through the use of flip-flops (e.g. registers and counters) and gates designed in a relatively ad hoc manner. Microprogramming the control unit has been proposed as an orderly alternative to this ad hoc design procedure where the hardware of the control section is replaced by a "microprogram control" unit.

Microprogramming is therefore a technique for implementing the control function of a digital computer using programmable control signals stored in a separate, word-organized memory, called control store. If the control store were a programmable memory, then the system architecture could be modified to optimize processing of each task to be performed; moreover, it could be altered completely to resemble the architecture of another, entirely different machine. The microprogramming of the control store of a computer system to alter the basic architecture enabling one

machine (the host machine) to execute machine language programs intended for another machine (the target machine) is defined as Emulation.

The remainder of this chapter is divided into three sections: general information on microprogramming, construction of an emulator and applications of emulation.

A. GENERAL CONCEPTS OF MICROPROGRAMMING

Since the cost of software has become a major portion of the overall cost of computer systems today, more and more manufacturers and users are turning to microprogramming the control section of their computer to tailor the machine to individual applications, thereby making computer programming more efficient.

The main function of the control section is to specify the conditions under which sets of gates are to be opened. In conventional machines this can lead to a very complex, hardwired system, especially if the instruction set is extensive. A relatively simple field change, addition or alteration, may require a major modification of this complex system. This is not true in a computer that uses a microprogrammable control store. The complex hardwired structure is replaced by microinstructions stored in the control store, one microinstruction per word, which tell the system which arithmetic and logic operations to perform by specifying which gates to enable/disable. In this way,

control of the computer is removed from the hardware and placed under the control of the microprogrammer. This places a heavy burden on the programmer since he must become intimately familiar with the innermost workings of the machine; however, it allows him to model the machine to his specific programming needs. This gives the microprogrammer extreme flexibility in his applications on the computer.

There are two types of microinstructions: vertical and horizontal. Vertical microinstructions usually control one operation; and the address of the successor microinstruction is implicitly the current address plus one, unless the current microoperation causes a branch. In horizontally microprogrammed machines each control bit of a microinstruction is assigned to a specific gate or set of gates. This means that one microinstruction can control multiple operations. On the other hand, vertical microinstructions are divided into separate fields, each of which are then decoded to enable/disable specific gates or sets of gates; therefore, horizontal microinstructions require less decoding and provide more flexibility. However, the advantage of vertical microinstructions is that they generally require shorter control words considerably reducing the overall cost of control micromemory. The length of control words vary from 20 bits (vertical) to in excess of 120 bits (horizontal); the average microinstruction is 50-80 bits. The D-Machine, the host machine for this thesis, utilizes a combination of

vertical (Type II) and horizontal (Type I) microinstructions in the form of a 56-bit word.

Since the execution time of one horizontal microinstruction is essentially the same as that of one vertical microinstruction, the multiple operations performed during one horizontal instruction is a desirable feature. This parallelism can drastically reduce the size of a microprogram, and dramatically decrease execution times. These reductions can only be realized, however, if the programmer can recognize concurrent actions and optimally group them into one instruction. A great deal of work has been done toward including the efficient use of available resources into a compiler for a high-level microprogramming language, but with little success. Either the probability of making the determination is low, or the cost of the optimization is too high. Much additional research is required in the area of recognizing and combining potentially concurrent actions.

Different microintructions specify different microoperations to be performed, and depending on the nature of the microoperations, some phase of one instruction may overlap a phase of a subsequent instruction. Instruction timing is, therefore, an important factor in the overall efficiency of microprograms in which microoperations are not mutually exclusive. A brief description of the timing considerations that must be made in microprogramming the Burrough's D-Machine is given in Chapter IV.

Many main-frame manufacturers hesitate to allow the novice programmer access to the innermost workings of their computer (e.g. registers, data paths, and gates). One method of prohibiting this access is to use read-only micromemory with the microprograms being provided by the computer manufacturer. This helps preserve the designed identity of the computer but does not provide for its optimum use. The programmer who is given the capability of modifying the logical design of the machine to his specific programming needs will find his programming tasks greatly simplified. For example, he could implement new instructions, such as floating point, which were not designed into the original machine.

In summary, microprogramming is becoming more widely used for the following reasons: 1) the flexibility and growth potential of microprogrammable machines (especially in the area of emulation), 2) increasing software costs and current semi-conductor technology favor microprogram design, and 3) user-oriented instructions can be designed into machines through microprogramming.

B. BUILDING AN EMULATOR

Emulation describes a process through which the hardware components of one machine (host) are made to "appear" to assume the specific characteristics of the hardware of another machine (target). This provides the host machine with the capability of executing machine language instructions written for the target machine. Simulation is a software process that provides the same capability for program execution; however, simulation involves interpretive execution by a high-level language program. Emulation differs in that an emulator performs its interpretive execution through the hardware. Simulations usually perform within a factor of 100-200 times real-time. Emulation is generally within a factor of 10, but often can be tuned to approach real-time. For this reason emulations are usually more cost-effective than simulations. The following paragraphs describe the process involved in building an emulator.

There are two major approaches to emulating a target machine: 1) the hardware and firmware (the physical realization of a microprogram) can be used in conjunction with a software simulator (software alone would be pure simulation), or 2) the hardware and firmware can execute with no software support.

Depending on the percentage of software utilized, a software-assisted emulation may be somewhat slower than a

firmware emulation. Typically, the programmer mentally develops a software simulation of the target machine. He then decides which sequences of instructions are most frequently used, and which are the hardest to simulate. These become candidates for microprogramming. Frequently a single new instruction can be microprogrammed to replace repetitive sequences of the same high-level instructions, thus speeding up the emulation. With this software-firmware approach the user must decide the point at which he must stop substituting microcode for software routines. This is generally dependent on the amount of control storage available, or the cost-performance criteria with which he is working. An example of a software-hardware approach to emulation is the IBM 7000 series emulators used with the IBM System/360 Model 65.

The Burrough's D-Machine provides the capability of emulation using the software-hardware method. A simulation is written in the high-level language ALGOL. ALGOL on the D-Machine allows the user to define a new operator which, when called in the simulation program, branchs to a preprogrammed series of microinstructions; executes the microinstructions; and returns to the ALGOL program. This gives the user the ability to execute frequently used routines (such as instruction fetch) from micromemory which has a faster cycle time than that of main memory where the ALGOL simulator resides. An advantage to this approach is that the system does not have to be regenerated after different emulators

are run since, in essence, only an ALGOL program has been executed. However, as previously discussed, this software emulator is slower than a firmware-controlled emulator.

The authors used the hardware-firmware approach in their emulation of the AN/UYK-7. This method will be developed in more detail along with suggested hardware additions which could enhance the emulation. Control of the system resides entirely in the firmware, which means that the emulator, once loaded, dedicates the machine to that emulation, and native mode programs can no longer execute until system regeneration. This unproductive overhead of loading and reloading the emulator and the native mode machine is one drawback. An additional drawback is the larger micromemory required since the entire emulation is microprogrammed. However, the emulator executes exclusively through firmware making this approach to emulation significantly faster. An example of a firmware-controlled emulator is the IBM 1401 emulator on the System/360 Model 30.

Although systems such as the D-Machine use horizontal microinstructions to enable parallel or concurrent operations, the microprogram can only perform one function of the target machine at a time. That is, if the target machine performs parallel operations such as executing one instruction while simultaneously fetching the next instruction, the microprogram can only emulate one of these functions at a time. It must execute the current instruction and then fetch the next instruction. For this reason, emulation is

usually slower than the real time performance of the target machine even though it may have a faster memory cycle time.

In order to emulate a target machine, the microprogrammer must first understand three areas: 1) the host machine, 2) the target machine and 3) the microlanguage. Second, the registers, counters, and accumulators of the target machine must be mapped into the host machine. This is often done in two steps - depending on the number of available registers in the host machine. The registers of the target machine most frequently used should be mapped directly to registers in the host where possible. The remaining registers of the emulated machine are mapped into the host's main memory. As many of the target machine's registers as possible should be mapped directly, so that fewer memory references will be required by the emulator. For example, if the emulator were executing a Load Accumulator instruction and the specified accumulator had been mapped into one of the host's registers, then only one microinstruction is needed to emulate the load instruction. However, if the accumulator had been mapped to a memory location, then it would take several microinstructions to emulate the entire operation - i.e. setup the store address, store the value to be loaded into the write register, and perform a write to main memory. Not only does this require additional instructions, but it also includes a main memory access, whose cycle time is slower than micromemory's cycle time. Some instructions may require two main memory cycles, such as an ADD where the

mapped accumulator must be read, added to and then written out. This demonstrates the necessity of mapping as many registers as possible to the host machine's hardware. Seldom, however, can this mapping be accomplished without some use of main memory; therefore, the microprogrammer must select some portion of main memory as a privileged area for register mapping. The authors reserved the first 1024 words of main memory in the emulation of the AN/UYK-7, for register and buffer mapping.

The next step in the emulation is to assume that a user's object program is resident in main memory. The emulator must then fetch, decode and execute the program, instruction by instruction. The microprogrammer must decide which emulation functions should be written as subroutines, and which functions should be written in line. The most frequently referenced subroutine in any emulation is the fetch routine since it is executed for every instruction. The normal steps involved in this routine are:

1. Determine the address of the next instruction as indicated by the program address register.
2. Read the instruction from the main memory address calculated in step 1. This is the current instruction to be emulated.
3. Decode the instruction into its designator fields.
4. Calculate the operand address. (The AN/UYK-7 uses a displacement plus an index register plus a base register).
5. Read the operand from main memory.

6. Perform indirection if allowed and indicated.

7. Some instruction sets, AN/UYK-7 for example, allow whole or half-words. Check if current instruction is half-word.

8. Some instruction sets operate on partial word operands. If so determine if quarter, half or full-word operand is to be used.

These steps must be executed for each instruction fetched from the user's program. Depending on the particular instruction being emulated, some of these steps may be omitted (step 8 is only performed for Format I instructions) but the majority are generally applicable. This demonstrates the overhead involved in emulation prior to branching to the microroutine that does the actual instruction execution.

Some hardware additions that can facilitate emulation are a fast shifter and a field select unit (FSU). The shifter simplifies variable-length byte extraction, a common emulation process. For instance, in the D-Machine it takes the same amount of time to shift a word by 1 bit as it does for any shift up to 31 bits. The FSU allows the testing of selected fields without requiring a cycle to go through the adder (the D-Machine does not have this feature). This relieves the programmer of masking and shifting fields prior to testing them. These two features are the most common and cost-effective additions to the hardware.

One approach to emulation that has recently come into use, but which is not that widely used or understood, involves using a combination of hardware, software and operating system. IBM is experimenting in this area, and has established the following design characteristics for the System/370 emulators:

1. Emulators must be integrated with the operating system and run as a problem program. This eliminates the overhead of loading and reloading the emulator and the native mode machines.
2. Allow multiprogramming of emulators as well as a mix of emulators and native mode programs.
3. All programs including emulators must be interruptible.

With such a system IBM felt they could give the user an improved, more efficient environment from which to operate, either in the emulator or native mode. The authors feel this should be a primary field for future applications.

The final item to be discussed in the Section is emulation of Input/Output operations. Most sources agree that emulation of I/O is as difficult, if not more so, than the implementation of the central processor's instruction set. This is true primarily because once the basic routines needed for instruction and operand fetch are programmed, the actual instruction executions are fairly easy to microprogram. This is not necessarily true with I/O - buffers must be set up; data conversion possibly performed; and perhaps the hardest point, emulating circumstances in which results are

not yet known. For instance, if the I/O is to search for a specified record, then what happens if the key cannot be found? There are many such checks or error conditions that make I/O difficult to emulate. Another major problem is emulating interrupt handling during I/O. Since emulation of I/O is essentially a separate topic and since the authors did not implement emulation of the AN/UYK-7 Input/Output instruction set as part of the thesis, I/O emulation will not be described in any further detail. However, references 5, 12, 15, 21, and 22 provide excellent material on this topic.

C. APPLICATIONS

This section is designed to familiarize the reader with some of the applicable areas for emulation usage. It is by no means an exhaustive study of this topic, nor is it intended to be so.

Emulation was first used in, and is still the primary application for, converting from second generation computing systems to third generation systems. Normally, the process is costly and dangerous in the aspect of converting programs running on the existing system to programs capable of executing on the new machine. There are several methods other than emulation capable of performing this task, none of which are completely satisfactory: simulation is slow; automated translation is unproven; and instruction by instruction reprogramming is costly and cumbersome. The advantage of emulation is that if the old machine could be emulated on

the new machine allowing direct execution of all old programs, then the old programs could be converted at leisure, if desired and justified. For example, reprogramming might not be cost effective or justified for a program with an expected life span of six months. In such circumstances continued emulation of the program is the solution. However, the situation might be different if the program had an expected life span of three years.

Another major application for emulation is the design, development, and testing of a newly conceived system on an existing system. The target machine does not necessarily have to be an existing computer system, and the user could experiment with the design of any machine he desires. In this way a richer instruction set can be achieved; moreover, software and diagnostic routines could be developed for the new machine prior to the machine even being produced or marketed. IBM used this method to some extent by emulating System/370 on System/360 prior to building System/370. This application tends to decrease the time from initial construction to marketing because software, such as the operating system, can be tested and debugged concurrently.

A third application, and one seen as a major trend for the future, is the ability of the user to adapt the computer to his individual needs. Modern-day computers are built as powerful, general-purpose machines designed to operate effectively over a large field of problems. In doing so, however, the machine cannot optimize execution of any one

particular application. Emulation gives the user this ability. The user can either operate from an existing emulator, or he can tune the emulator to his individual application through microprogramming. One user may want a machine oriented toward string operations, while another user may be doing extensive scientific calculations. If each user could execute from a separate emulator tuned to his individual application, then each would be more efficient and productive. In addition the computer would probably be easier for the user to program. This is the area where multiprogramming of emulators is projected to be of the most use.

There are several other areas of emulation usage which were not discussed; such as the academic environment for research purposes (the reason for this thesis), and tuning of existing software. However, as previously stated, this section was designed only to give the reader a flavor of the possible uses of emulation and perhaps stimulate his own ideas for possible applications. For additional and more detailed material see references 2, 15 through 18, and 20 through 24.

III. THE AN/UYK-7

This chapter is intended to introduce the reader to those operational characteristics and capabilites of the AN/UYK-7 computer most pertinent to the Emulation. This will assist the reader in understanding the design and mappings used in this Emulation and presented in subsequent chapters. This chapter is not intended to be a technical manual and should not be used for that purpose. The reader should consult references 26 and 27 for more detailed information.

The AN/UYK-7 is a highly reliable, ruggedized, multiprocessor system designed and manufactured by the Univac Division of Sperry Rand Corporation. Built to military specifications (MIL-E-16400), it is utilized extensively throughout the United States Navy in Tactical Data Processing applications [26]. Rapid data transfer rates are provided during communications between external devices, internal random-access memory, and the central processor. The system is capable of average command execution times of 1.5 microseconds, and an input data transfer rate of over one million 32-bit words per second. The AN/UYK-7 maintains two completely separate sets of index, arithmetic and relative address (Base) registers, for use during Task and Executive states. In addition, a set of 18 privileged instructions

with special characteristics for executive control are reserved for the Interrupt State. These features along with the parallelism of AN/UYK-7 field and register references, make it an extremely difficult system to emulate in real-time.

The AN/UYK-7 was designed to operate as either a single or multiple processor system with up to 256K, 32-bit words of random access memory. This allows on-line access to 1024K bytes through an instruction feature which permits whole-word, half-word, or quarter-word memory references. This Emulation assumed an AN/UYK-7 configuration consisting of a single processor, monoprogramming, with one 64K, 32-bit word memory module. Partial word references were fully emulated providing random access to 256K bytes.

A. INTERRUPTS

The AN/UYK-7 processes data in real-time in response to a highly prioritized interrupt system with decision-making properties. This priority system allows the central processor to select that program routine which is necessary to respond to the interrupt requiring the most urgent attention. Since the Emulator was designed for a monoprogramming environment using asynchronous I/O, the interrupt features of the AN/UYK-7 were not fully implemented. Several types of interrupts, such as Program Faults (Illegal Instructions), were implemented and tested. All interrupt handling flags, lockouts and registers were mapped into the Emulator.

This will facilitate full implementation of interrupt features - should a decision be made to incorporate multiprogramming or synchronous I/O in the course of expansion to a full Emulation.

B. CENTRAL PROCESSOR

The AN/UYK-7 central processor contains all the control, arithmetic and timing circuitry required for processing alphanumeric data and for executive functions. The central processor operates in two different modes or states: the Interrupt State executes executive-type functions, and the Task State processes user programs. A group of 16 privileged instructions and a separate set of arithmetic, index and base registers are reserved for the exclusive use of the processor while in the Interrupt State. These features greatly enhance the AN/UYK-7's multiprocessing and multiprogramming capabilities. These special instructions and registers were mapped into the Emulator; however, they were not fully implemented because of the monoprogramming environment in which the Emulator was to be run.

The AN/UYK-7 maintains a central processor control memory (CMR) consisting of 82 integrated-circuit, random-access registers of varying sizes appropriate to their function (e.g. A-registers 32-bits, B-registers 20-bits). The various registers are grouped into stacks according to their use, addressing, and relative size. In the Emulator, these registers were mapped into the D-Machine's main memory

starting at address 0000, with address assignments corresponding exactly to that of the AN/UYK-7. The notable exception was that CMR addresses designated as "Unassigned" in the AN/UYK-7 were used as temporary storage registers in the Emulator. Register mapping is discussed in more detail in Chapter V and presented in tabular form in Figure V-2.

The AN/UYK-7 registers of particular interest to the user are: the Program Address Register (PAR); the Active Status Register (ASR); and the Base, Index and Arithmetic Register groups. These are the only registers which are either directly addressable or accessible to the programmer.

1. Program Address Register (PAR)

The PAR is a 20-bit register used for addressing the next instruction to be fetched from memory for execution. It consists of a 16-bit displacement value and a 3-bit Base Register reference. The effective instruction address is formed by the addition of the displacement and the contents of the selected Base register. The PAR displacement value is incremented by one word at the completion of each instruction cycle with the exception of JUMP instructions. These instructions cause replacement of the PAR by the "s" and "y" operand fields of the JUMP, thus providing for out-of-sequence instruction execution. The PAR mapping in the Emulator is not composed of two separate fields, but rather as the calculated effective address. This greatly simplifies and expedites instruction references; however, it

causes some difficulty with specific instructions which require access to the two separate fields of the PAR. When encountered, this problem was overcome by dividing the PAR value by 8K. Since the Base registers were preinitialized in 8K increments, starting at 1024, the results of the division yielded a quotient equivalent to the Base register assignment, and a remainder equal to the displacement.

2. Active Status Register (ASR)

The ASR is a 23-bit register used to indicate and control the status of various operations in the central processor. Individual bits of the register are assigned special functions, and when set indicate that a particular status exists and that the processor should be controlled accordingly. Individual bits are set/cleared as the result of either an instruction execution or direct processor intervention. The bits of particular interest to the programmer are the arithmetic bits (equal, greater than or equal, overflow, and limits) which are set as the result of arithmetic functions, and may be interrogated by specific instructions, such as conditional jumps. Based on the condition of the bit tested these conditional instructions may cause a change in the program sequence. In the Emulator the ASR bits most frequently referenced by the programmer are mapped into a D-Machine internal register together with the PAR. This considerably reduces main memory references, and reduces Emulator execution time. The ASR mapping is discussed in detail in Chapter V.

3. Base Register (S)

There are two sets (Interrupt, Task) of 18-bit S-Registers, numbered 0-7. These registers, used in final Y-Operand and instruction address calculations, are necessary in a multiprogramming and multiprocessing environment. In spite of the monoprogrammming environment of the Emulator, the registers were mapped and used as designed; however, they were preinitialized by the Loader at 8K boundaries starting at address 1024. This provided the Emulator access to 64K of main memory in increments of 8K pages.

4. Arithmetic Register (A)

There are two sets (Interrrupt, Task) of 32-bit A-Registers, numbered 0-7. They are used extensively throughout the instruction set to hold one or more of the operand - inputs to, or results of arithmetic functions. The A-Registers also serve as the main interface between the central processor and main memory. These registers are directly addressable by the programmer.

5. Index Register (B)

There are two sets (Interrupt, Task) of 20-bit B-Registers, numbered 1-7. These registers can be used as counters, or they can be employed in a special addressing technique called Indexing. During normal Y-Operand address calculations, the designated B-Register is added to the y-displacement and a specific S-Register to form an effective

address. This address is then considered to have been indexed by the B-Register value. Indexing provides the programmer with the option of sequentially referencing memory by merely incrementing the index during each pass through a loop. Reference to any B-Register other than B(0) implies that indexing is to take place during the Y-Operand address calculation. A reference to B(0) is treated as an index of zero since B(0) is not a valid register. In the Emulator, the contents of B(0) is always zero, and references to B(0) are conveniently treated the same as references to B(1) thru B(7). In the AN/UYK-7, the B-Registers have the same internal structure as the PAR; that is, they consist of two fields (a Base register, and a displacement). In the Emulator, B-Registers are treated as consisting of one field which can be separated into its two respective fields by division by 8K, as previously described in the PAR discussion.

In order to minimize access time and field decoding, all CMR registers were treated as 32-bit registers in the Emulator. This caused no problems with the exception of the PAR and Index Registers, which were resolved as previously discussed. The A, S, and B-Registers are directly addressable through the Load/Store CMR Instructions of the AN/UYK-7. They are incidentally addressable through references in specific fields of the different Format instructions reserved for that purpose.

C. INSTRUCTION FORMAT

Instructions of the central processor appear in five different formats according to their operational characteristics, as presented in Figure III-1. Formats I, II, and III occupy a full, 32-bit computer word; Formats IV-A and IV-B each occupy a half computer word. Two half-word instructions can be stored in one memory location. When a half-word instruction in the upper half of the computer word is executed, the processor sets bit 15 of the ASR; if a whole-word or lower half-word is executed the bit is cleared. In the Emulator, this bit determines whether the subroutine IFETCH or HALFFETCH is executed. IFETCH reads the next 32-bit instruction from memory. HALFFETCH shifts the lower half-word of the current instruction into the upper halfword position for execution.

Each of the five formats divide the instruction into different fields. Each field except "y" (the constant or address field) has a particular function in controlling the various internal enables and commands required for proper execution of the instruction. Some fields define the use, modification or application of the y-field to secure the desired operand; others select an accumulator, index, or base register; others select an IOC, define a sub-function code, or combine to form a special interpretation defined by the instruction. The AN/UYK-7 maintains a repertoire of 132 central processor instructions whose specific functions are defined in detail in reference 26.

Bit No.	31--26	25-23	22-20	19-17	16	15-13	12---0
FORMAT I	f	a	k	b	i	s	y
FORMAT II	f	a	f2	b	i	s	y
FORMAT III	f	a	f3/k	b	i	s	y

Bit No. Bit No.	31--26 15--10	25-23 9-7	22-20 6-4	19-17 3-1	16 0
FORMAT IV A	f	a	f4	b	i
FORMAT IV B	f	a	m		

INDIRECT CONTROL WORD FORMATS (Indirect Addressing)

Bit No.	31-30	29-25	24-20	19-17	16	15-13	12---0
	c	w	p	b	i	s	y
	c	c1	unused	b	i	d	

Elements of the word are interpreted as follows:

FIELD	BASIC DEFINITION
f	6-bit function code
f2	3-bit subfunction code
f3	2-bit subfunction code
f4	3-bit subfunction code
a	3-bit accumulator register designator
k	3-bit operand interpretation designator
m	6-bit shift count designator
b	3-bit index register designator
i	1-bit indirect addressing designator
s	3-bit base designator
y	13-bit address displacement/operand designator
c	2-bit control designator
c1	1-bit indirect subfunction designator
w	6-bit character length designator
p	5-bit position indicator for character length
d	16-bit address displacement

INSTRUCTION AND INDIRECT ADDRESS WORD FORMATS [27]

FIGURE III-1

Of particular note to the reader is the K-Field of the Format I instructions. The value of K designates whether the operand to be fetched/stored is a whole, half, or quarter-word operand, as depicted in Figure III-2. If the value of K implies a partial-word operation, then it also determines which portion of the 32-bit word is to be accessed. That is, a K-value designating a quarter-word (byte) transfer can also specify that the byte is located in the upper, lower, or one of the intermediate two bytes of the 4-byte, 32-bit operand. It is this function that makes the AN/UYK-7 an extremely powerful word processing machine. This function was fully implemented and tested in the Emulator.

K	READ MEMORY to ARITHMETIC	STORE ARITHMETIC to MEMORY
0	sy(SE)+B(b) -> A15-0(SE)	NOT USED
1	Y15-0 -> A15-0 (SE)	A15-0 -> Y15-0;Y31-16 unchg
2	Y31-16 -> A15-0 (SE)	A15-0 -> Y31-16;Y15-0 unchg
3	Y31-0 -> A31-0	A31-0 -> Y31-0
4	Y7-0 -> A7-0 (ZE)	A7-0 -> Y7-0;Y31-8 unchg
5	Y15-8 -> A7-0 (ZE)	A7-0 -> Y15-8; Y31-16 unchg Y7-0 unchg
6	Y23-16 -> A7-0 (ZE)	A7-0 -> Y23-16;Y31-24 unchg Y15-0 unchg
7	Y31-24 -> A7-0 (ZE)	A7-0 -> Y31-24; Y23-0 unchg

SE--sign extended; ZE--zero extended
A is the ACCUMULATOR specified by the a-field

FORMAT I INSTRUCTION K-FIELD INTERPRETATION [27]

FIGURE III-2

D. MODES OF OPERATION

There are several different modes of operation of the AN/UYK-7; most of them are concerned with different addressing techniques. Specifically, they are: normal mode, index mode, repeat mode, and indirection. In the normal mode, the Y-Operand address is calculated as the sum of: the y-offset, a base register and an index register, or Y=y+B(b)+S(s). There are two exceptions to this general formula. First, if the K-field of Format I instructions is a zero then Y=sy+B(b), where "sy" is the concatenation of the s-field and y-field of the instruction. Second, for all instructions regardless of Format, if the B-field is zero then the B(b) value used in calculating the Y-Operand address is always zero. There are several additional addressing techniques employed during the repeat and indirect modes of operation. They are described in detail in Chapter V. These ad hoc addressing techniques employed by the AN/UYK-7 greatly enhance its capabilities but were extremely difficult functions to emulate.

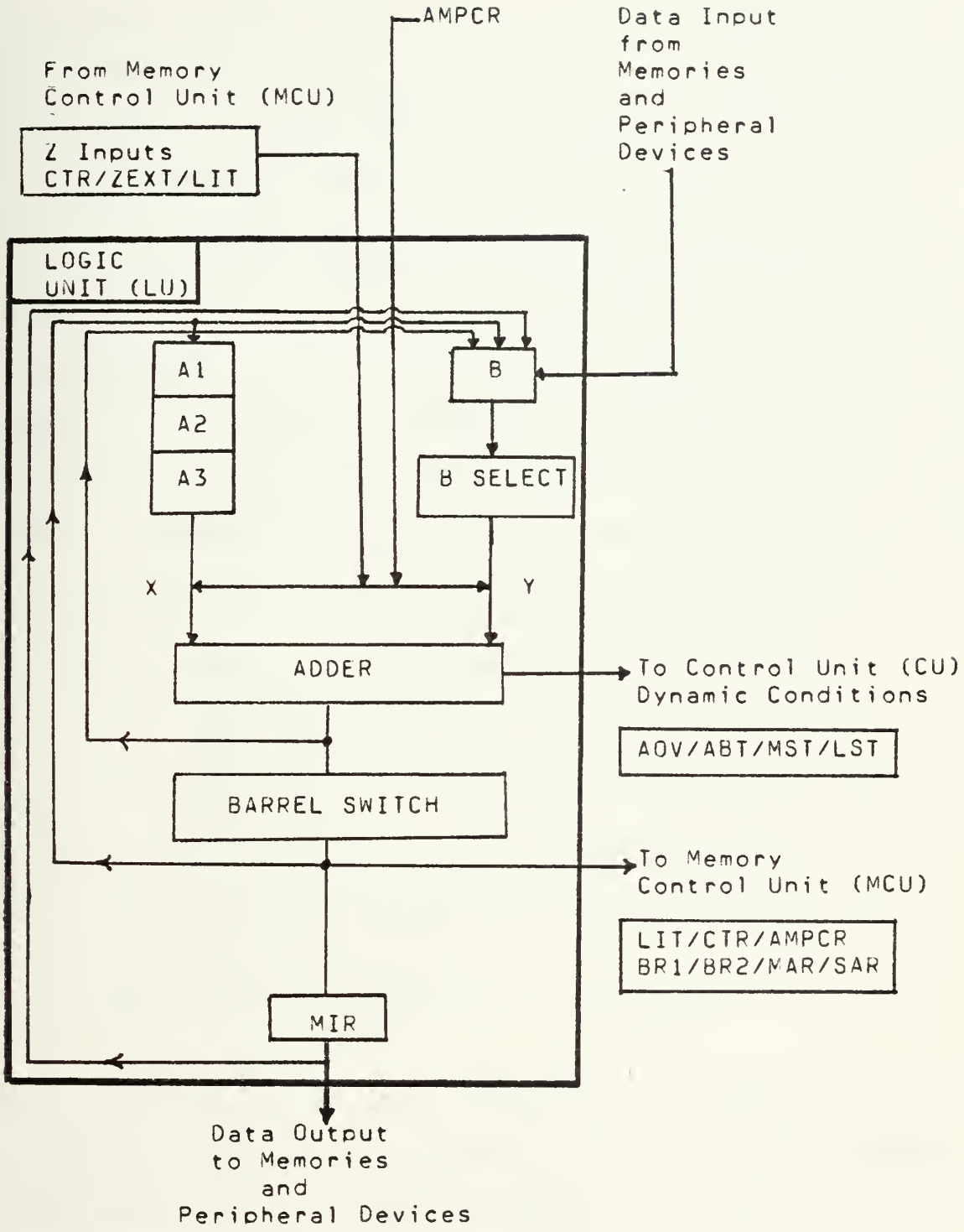
IV. BURROUGH'S D-MACHINE

A. GENERAL DESCRIPTION

An Interpreter Based System is a digital processing concept developed by Burrough's Corporation, that utilizes hardware building blocks which can be tailored through microprogramming to perform as a variety of general purpose or special purpose general processors [7, 8, 24]. The basic Interpreter, also referred to as the D-Machine, is the primary building block of this unique system. It is composed of five functional modules each of which can be modified to manipulate an 8 to 64-bit word format in increments of 8 bits. In the Naval Postgraduate School configuration, two of these modules, the Nanomemory and the Micromemory, are combined into one. The register sizes given in the following descriptions of the modules are specific to this configuration which is a 32-bit D-Machine.

1. Logic Unit (LU)

The Logic Unit (LU), presented in Figure IV-1, contains all the registers, the Barrel Switch and the Adder which are available to the microprogrammer for manipulating data fields during the process of emulating an instruction. A description of each of these registers and its functions follows [8,24].



LOGIC UNIT (LU) BLOCK DIAGRAM

FIGURE IV-1

The 32-bit registers A1, A2, and A3 are functionally identical. They temporarily store data within the Interpreter and serve as the primary (X) input to the adder. Any A-Register can be designated as a destination for the output from the Barrel Switch.

The 32-bit B-Register is the primary external input interface (from the Switch Interlock). It also serves as the secondary (Y) input to the Adder, and can receive the output of the Adder, in addition to the Barrel Switch which is the normal source for the results of arithmetic calculations. The B-Register can be the destination of any of the following during execution of any one microinstruction:

a. The Barrel Switch output.

b. The Adder output.

c. The external data from the Switch Interlock.

d. The Memory Information Register (MIR).

e. The carry complements of four-bit or eight-bit groups.

f. The Barrel Switch output ORed with b, c, or d above.

The output of the B-Register has true/complement selection gates which are controlled in three separate sections: the most significant bit (MSB), the least significant bit (LSB), and all the remaining internal bits. Each of these parts is controlled independently, and may be either all one's, all zero's, the true contents or the one's complement of the respective bits of the B-Register.

The 32-bit Memory Information Register (MIR) buffers the information which is to be written to main system S-Memory, or to a peripheral device. It is loaded from the Barrel Switch output and its output is directed either to the Switch Interlock or the B-Register.

The 32-bit Adder in the LU is a modified version of a straightforward carry look-ahead adder. Inputs to the Adder are from selection gates which allow various combinations of the A, B, and Z inputs. The A input is from the A-Register output selection gates and the B input is from the B-Register true/complement selection gates. The Z input is an external input to the LU and can be:

a. The output of the counter in the Memory Control Unit (MCU) into the most significant eight bits with all other bits being zeros.

b. The output of the literal register in the MCU into the least significant eight bits with all other bits being zeros.

c. An optional input into the middle bits with the most and least significant bytes being zeros.

d. All zeros.

There are always two inputs to the Adder referred to as the X-input and Y-input. An X-input may have A, Z, or zero as its source. A Y-input may have B, Z, or one as its source. An unspecified X-input is always assumed to be zero. Using various combinations of the possible inputs to the selection gates, any two of the three (A, B, or Z) inputs can be added together, or can be added together with an

additional one added to the least significant bit. In addition, all binary Boolean operations can be performed between any two inputs.

The 32-bit Barrel Switch is a matrix of gates that shifts a parallel input data word from the Adder, any number of places to the left or right, either end-off or end-around. The output of the Barrel Switch may be gated to one or more of the following simultaneously:

a. The A-Registers (A1, A2, A3).

b. The B-Register.

c. Memory Information Register (MIR).

d. Least significant 16 bits of MCU registers (BR1, BR2, MAR, AMPCR, CTR).

e. Least significant 5 bits to the Control Unit (CU) for the shift amount register (SAR).

2. Memory Control Unit (MCU)

One MCU is required for an Interpreter to have access to 64K of main memory [8, 24]. The addition of a second MCU is possible, thus expanding on-line memory access to 128K. The MCU has three major sections:

a. The microprogram address section contains a microprogram count register (MPCR), the 12-bit alternate microprogram count register (AMPCR), the incrementer, the microprogram address controls register, and their associated control logic. This section is used to address the Microprogram Memory (MPM) for the sequencing of microinstructions. The AMPCR contents may be used as a Y-input to the Adder.

b. The memory/device address section contains the 8-bit memory address register (MAR), the two 16-bit base registers BR1 and BR2, the output selection gates, and their associated control logic.

c. The Z register section contains registers which are the Z inputs to the LU Adder: a loadable counter (CTR), the literal register (LIT), selection gates for the loadable counter and their associated control logic.

3. Control Unit (CU)

The Control Unit (CU) has five major sections: the shift amount register (SAR), the condition register (COND), part of the control register (CR), the MPM content decoder and the clock control [8, 24].

The functions of the SAR and its associated logic are:

a. To load shift amounts into the SAR for use in (0 to 32-bit) shifting operations.

b. To generate the required controls for the Barrel Switch to perform the shift operation indicated by the current microinstruction.

c. To generate the "word length complement" of the SAR contents, where the "complement" is defined as the amount that will restore the bits of a word to their original position after an end-around shift of N followed by an end-around shift by the "complement" of N.

The control register (CR) is a 56-bit register that stores all those control signals from the current microinstruction which are not used in phase I. The CR is divided into the phase III controls and the MPAD controls.

The condition register (COND) section of the CU performs four major functions:

a. Stores 12 resettable condition bits in the condition register. The 12 bits of the condition register are used as interrupts, error indicators, status indicators, and lockout indicators.

b. Selects 1 of 16 condition bits; 12 from the condition register and 4 dynamically generated during the present clock time in the Logic Unit for use in performing conditional operations.

c. Decodes bits from memory for resetting, setting or requesting the setting of certain bits in the condition register.

d. Resolves priority among Interpreters in the setting of global condition (GC) bits which provide a facility for inter-Interpreter lockout control.

4. Microprogram (M-Memory)

The Micromemory (M-Memory), also known as the Nanomemory (N-Memory) in the Naval Postgraduate School configuration, is a programmable control store memory which contains the user supplied microprograms in the form of microinstructions. Each microinstruction consists of a 56-bit nanoinstruction and provides the gating for functioning of the previously discussed LU, MCU, and CU modules. Micromemory consists of two 4K, 56-bit, modules allowing the

programmer access to approximately 8K of control store. The sequencing of microprogram instructions is controlled by the following procedure: the Nanomemory provides information to the condition testing logic indicating which condition is to be tested. The condition testing logic provides a TRUE/FALSE signal to the successor logic which selects between the three TRUE and three FALSE successor bits. These three successor bits provide eight possible successor command combinations which are listed below with their associated interpretations:

a.	WAIT	Repeat the current instruction
b.	STEP	Step to the next Instruction
c.	SKIP	Skip the next instruction
d.	JUMP	Jump to address in AMPCR
e.	RETN	Return from a micro subroutine
f.	CALL	Call a micro subroutine
g.	SAVE	Save the address of the head of a loop.
h.	EXEC	Execute one instruction out of sequence.

The particular successor command specified then provides the controls needed in the selection (MPCR/AMPCR) and incrementing logic to generate the next MPM address. Except for the EXEC command the MPCR is loaded with the MPM address.

For a more thorough description and discussion of these five modules, reference 10 provides an outstanding

diagrammatic breakdown of the Interpreter's internal hardware configuration and operation. Reference 7 discusses the interface of various Interpreter, peripheral, and memory configurations that have been tested by the Advance Development Organization of Burrough's Defense Space and Special Systems Group.

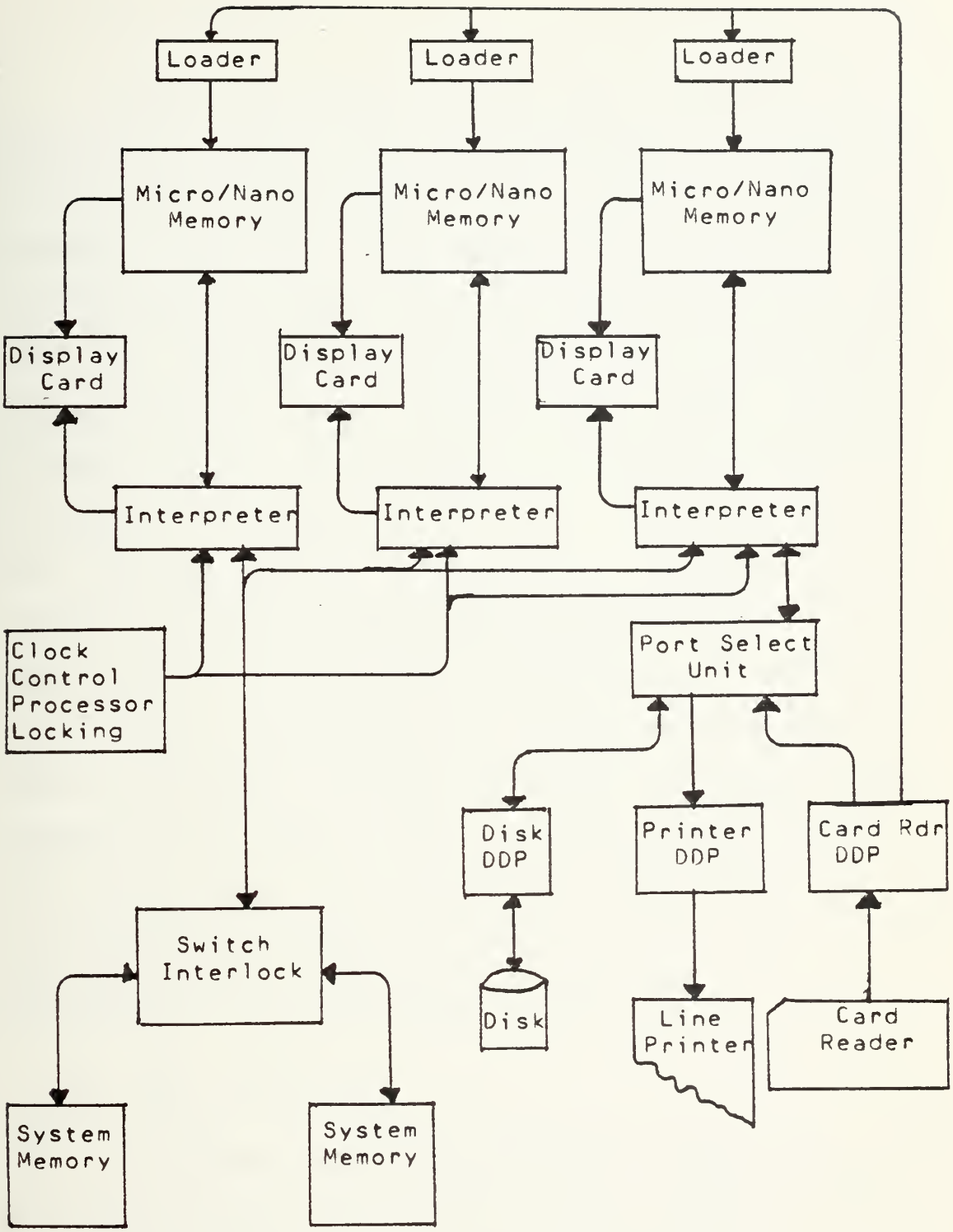
In general, there are three data processing functions to which the Interpreter may be applied: emulation of existing or hypothetical machines; direct execution of high level languages; and problem solution through microprogram "tuning" of the machine to the problem. The initial goal for the installation at the Naval Postgraduate School is emulation of existing machines such as the AN/UYK-7, with further expansion and development as a teaching tool in the areas of operating systems, file management, compiler writing, and emulation of hypothetical systems.

B. NAVAL POSTGRADUATE SCHOOL CONFIGURATION

1. Description

The system presently installed at the Naval Postgraduate School consists of three Interpreters, one 64K memory module, a card reader, line printer and dual cartridge disk. Only one of the three Interpreters is allowed to communicate directly with any peripheral and is discussed below under I/O Interface. Figure IV-2 represents the present configuration; however, future expansion calls for the addition of a supervisor's console and a cross-connection with the school's PDP-11/45. This will enhance the attached peripherals by three tape drives and greatly increase the amount of on-line storage to the point where implementation of a CMS-2 Compiler may be feasible.

When the system is energized and initialized daily, its basic microstructure is that of two B-6700 LIFO ALGOL Stack Machines - each with 32K of memory - attached to a single Input/Output Processor. The system is capable of pseudo-multiprocessing in this configuration. ALGOL programs submitted through the card reader are executed in a batch mode with the two processors competing to fetch the next job. All utiltiy programs, the operating system and the file manager are executed with the machine in this configuration. In order to change the machine's configuration, the user must alter the microcode in order to execute the desired machine such as, the AN/UYK-7.



NAVAL POSTGRADUATE SCHOOL THREE INTERPRETER SYSTEM

FIGURE IV-2

2. I/O Interface

Since only one Interpreter (IOP) is allowed to exchange data directly with the peripherals, it is necessary for the remaining two processors to communicate their I/O requests to the IOP. This is done by the Interpreter placing a message in address 64K otherwise known as the "Mailbox", and then issuing an interrupt (INT) to the IOP. The IOP periodically tests for INT and when sensed, reads the Mailbox, decodes the message, and performs the predefined function. After completing its task, the IOP places a message in the Mailbox informing the requesting Interpreter whether or not the I/O was performed successfully. The IOP then issues an INT to the Interpreter which sent the request. When the requesting Interpreter receives the INT, it reads the Mailbox to determine whether its request was performed and continues accordingly. This method allows the two Interpreters to act completely independently, each referencing only its assigned segment of memory and the IOP performing all I/O asynchronously upon request.

3. Memory Interface

The memory module consists of a single ported, 64K, 32-bit word, core memory, which is the equivalent of 256K of on-line, 8-bit character storage. Access to this single memory is through a Switch Interlock Unit (SWI) which makes memory appear to be multiported to the user. The SWI acts on a priority basis with that Interpreter having the lowest

number (the IOP) given the highest priority. Once an Interpreter issues a memory read/write reference, it may continue execution and need not wait on a memory completion signal. However, the memory address register (MAR) on a Read/Write, and the Memory Information Register (MIR) on a write, should not be changed until a completion signal is received. Thus a microprogrammer who anticipates his memory accesses and intersperses these instructions among the other code, should never have a delay caused by memory referencing.

C. INSTRUCTION TIMING

The Interpreter uses a one megahertz clock and initiates a new microinstruction every microsecond. The Interpreter enhanced with Emitter Coupled Logic (ECL) and a higher speed memory can operate off an 8 Megahertz clock. A corresponding 8-fold improvement in execution times can be expected.

There are two basic instruction types in the Interpreter. Type I uses two phases (I and III) for execution; although, its phase III may be held in abeyance until completion of a subsequent Type II, which always uses only phase I to complete. Phase I of the following Type I instruction always overlaps Phase III of the previous Type I. Type I instructions involve condition testing, external functions and Adder/Logic operations. Type II instructions involve literal assignments to one of three registers (LIT, SAR and AMPCR). Appendix D of reference 9 contains an excellent discussion of instruction timing.

D. LANGUAGES

The D-Machine has two resident programming languages: the language of the operating system, ALGOL; and the microprogram assembly language, TRANSLANG.

1. ALGOL

The resident programming language of the Burrough's D-Machine is the ALGOL 60 Language enhanced with additional language constructs which permit manipulation of data in the form of character strings. ALGOL employs a vocabulary of reserved words and symbols. The structure of the language, syntax, reserved words and symbols, and additional features of the ALGOL implemented on the D-Machine are discussed in reference 10. The accepted character set for ALGOL varies depending on the machine used and the character set or sets available on the machine. The Naval Postgraduate School configuration requires that all characters be converted from EBCDIC into the 6-bit Burrough's Common Language (BCL) format for recognition by ALGOL. This conversion is performed by the IOP when it is used for interfacing communications from external devices to the other Interpreters. When information is directed to an external device the IOP performs a reverse conversion, from BCL to ASCII. Thus all inputs to the emulator from peripherals are 8-bit characters containing a 6-bit BCL code. The Interpreter's file management routines, operating system and utilities are written in ALGOL, for execution on the ALGOL Stack Machine

configuration. ALGOL source modules may be inputted to the ALGOL Compiler from disk or cards. Object modules are placed in user libraries on disk for future execution by the stack machine. The operating system (MCP) will load specific object modules for execution when it recognizes a "?RUN filename" control card, where filename is assumed to be a precompiled ALGOL program.

2. TRANSLANG

The microprogrammer is aided in producing microprograms by a Microtranslator/Assembler that translates symbolic instructions written in TRANSLANG into microinstructions. Reference 9 describes the structure of TRANSLANG by defining its syntax and semantics, and presenting a series of examples. The reference also includes descriptions of register state changes resulting from executing microinstructions. Interpreter controls and timing are explained. Coding techniques and conventions are discussed via sample programs. The reference analyzes external operations with main memory and peripheral devices, and the coordination and control of multiple Interpreters via the Switch Interlock and global condition bits.

The Microtranslator is written in ALGOL for the D-Machine. It is written modularly with each function setup as a procedure call. The language, TRANSLANG, used ALGOL as a model; however, almost the entire language is composed of reserved words. Reserved words have very specific meaning

to the translator and cause specific nanoinstructions to be developed. TRANSLANG is free form and each instruction may be written in almost any order; however, multiple instructions appearing on the same line or card must be separated by a period. Each TRANSLANG instruction corresponds to one microinstruction, which is the set of Interpreter functions performed in parallel at each machine clock. The constructs provided include iterative mechanisms, I/O, Boolean, logical and computational operations, control transfers, and assignment functions. In order to provide control points for transfer operations, each instruction may be labeled with a symbolic M-Address. The output module of the Microtranslator is placed on disk and when loaded into micromemory, alters the basic machine configuration. Reference 9 is used as the microprogrammer's programming manual; although, there have been several enhancements to the machine and language which are not included in the manual. These modifications are discussed in Appendix A of this thesis.

V. PROJECT DESCRIPTION

The AN/UYK-7 achieves its speed and versatility partially through concurrent field utilization and partially through the provision of special and general purpose registers. Its emulation is complicated by the various modes of instruction operation (repeat, indirection, indexing) and in particular by the use of ad hoc addressing techniques. The fact that the AN/UYK-7 performs many of its operations in parallel makes it difficult for an emulation which is imitating one facility at a time to run in AN/UYK-7 real-time. For the remainder of this discussion the acronyms A(a), B(b) and S(s) refer to the Accumulator, Index and Base registers, respectively. The subscripts (a, b, and s) correspond to the register specified in the cognizant field of the instruction, and may assume values of 0 to 7. Capitol "Y" refers to the effective 18-bit, operand address; small "y" refers to the 13-bit, y-field of the instruction. The combination sy-field is formed by concatenation of the 3-bit, s-field and 13-bit, y-field of the instruction and forms an alternate operand address.

During the initial analysis of the Emulation it was decided to separate the project into the following phases:

First, the machine would be designed to accommodate a full emulation including multiprogramming and all IOC

functions. This required that all registers and condition bits of the actual AN/UYK-7 hardware be mapped into the model, even if they would not be used during this partial Emulation.

Second, it was decided to divide the Instruction Repertoire into the following groups:

a. Those instructions that could be emulated immediately and tested completely; such as, all Format I's and II's except Multiply, Divide, Square Root and Double Register Operations.

b. Those instructions that could be done prior to completion but for which there might be insufficient time for complete testing; such as, all Format III's, Shifts, Multiply, Divide, and Double Register Operations.

c. Those special instructions and functional modes that would be incorporated if at all possible; such as, indirection and repeat.

d. Those features for which there would definitely be insufficient time for incorporation; such as, Interrupts, IOC Instructions, and Floating Point Operations.

Third, a "Loader" program which could read AN/UYK-7 instructions from cards, decode them and place the results into memory for execution by the Emulator was mandatory.

Fourth, some facility for monitoring and debugging each instruction as it is executed by the Emulator would be required.

Fifth, some facility to output the results of each AN/UYK-7 program and to input test data was needed.

To accomplish these five tasks, two basic programs called the "Loader" and the "Emulator" were written. Each of these programs occupied its own Interpreter whenever the system was reconfigured as an AN/UYK-7.

A. LOADER

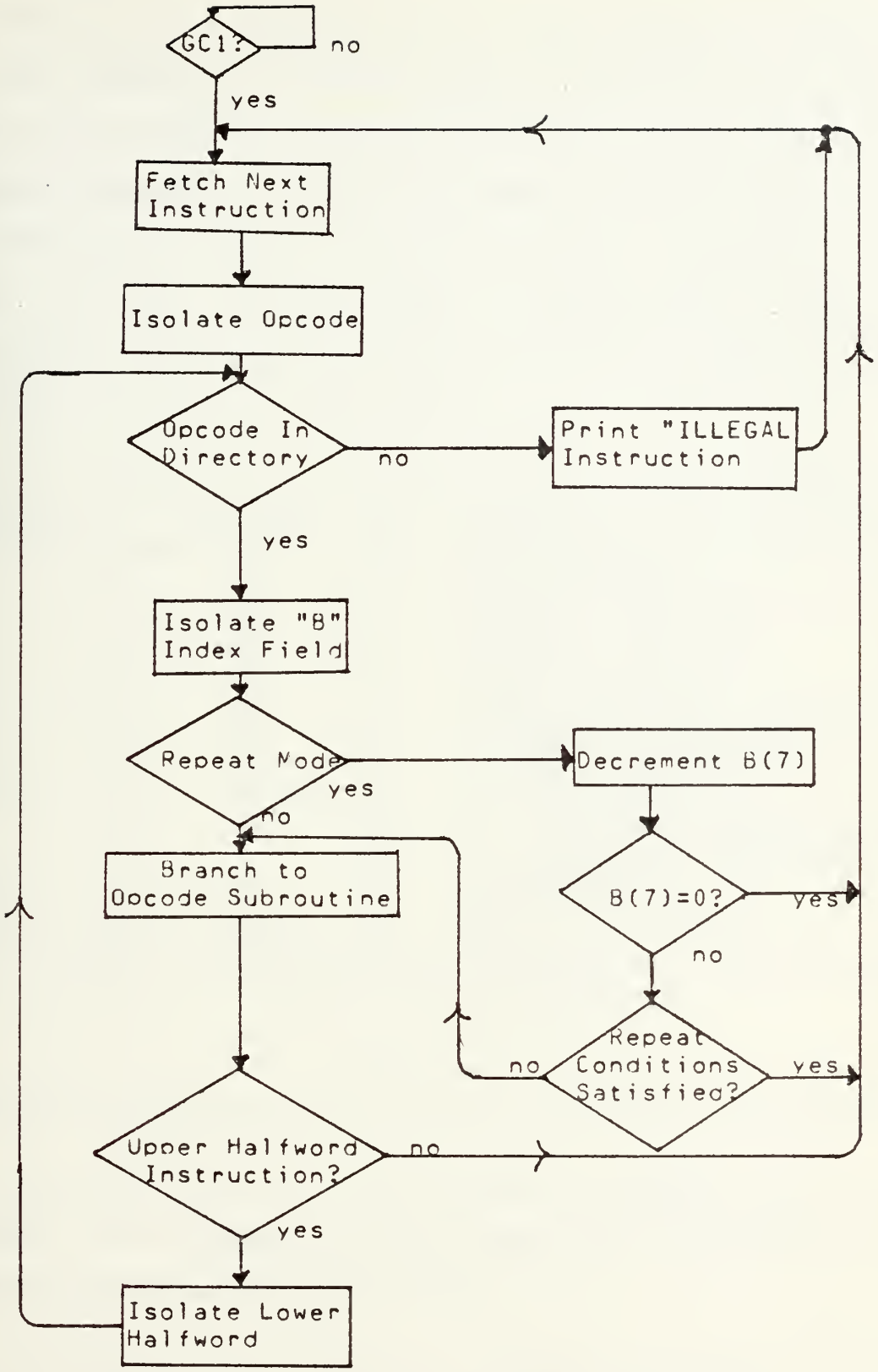
The Loader was written in TRANSLANG and fulfilled the third, fourth and fifth requirements; that is, it served as a combination loader, debugger and IOC monitor. Its use is described in Appendix A, the User's Manual section of this thesis. In general, the loader portion of the program acts like a pseudo-assembler. It has macros for defining a constant or character string, for saving space, for intitializing registers and switches, and for inputting instructions. The loader will accept one instruction per card and determine whether it is a Format (I, II, III, IV-A, or IV-B) instruction. Based on this determination, the Loader will decode the specified fields from the card and generate a binary instruction which is placed in the next sequential location in the user's memory. Upon detecting an "N" card, the loader portion of the program signals the Emulator - which is resident in the alternate Interpreter - that instructions are available for execution.

At this point the loader function ceases and the program can be called upon by the Emulator to act as a debugger or as an IOC. As a debugger, the Loader can be used to generate a number of error messages to the operator, or to provide an actual dump of control memory registers 000 to 035, which includes all Task Registers, the PAR and the Active Status Bits. As an IOC, the Loader can be called upon to read a card, print a buffer, or read the disk.

It should be emphasized that the Loader was written strictly as a tool to assist in the development and testing of the Emulator. Presumably, once a full emulation is written and tested, the IOC functions will be incorporated into the Emulator, while the loader and debugger functions will be either written in AN/UYK-7 Machine Language, or absorbed by the Emulator. When this is done, the Interpreter previously utilized by the Loader will be free for loading as a second AN/UYK-7, and a true multiprocessing environment could be created. A copy of the Loader is provided in Appendix C.

B. EMULATION PROGRAM

The Emulator was written in TRANSLANG microcode, and is loaded directly into the Interpreter's micromemory from disk. A copy of the Emulator is provided in Appendix B. Overall program flow for the Emulator is presented in Figure V-1.



OVERALL PROGRAM FLOW

FIGURE V-1

Besides the necessary Control Memory Register mappings, it was necessary to minimize memory references by maintaining certain frequently used information in the D-Machine's internal registers. For this reason, the PAR is maintained in the lower 20 bits of the D-Machine's A2 register. The upper 12 bits of A2 are used to maintain some of the most frequently referenced bits of the AN/UYK-7's Active Status Register as listed in Table V-1.

ASR Bit	A2 Bit	Function
2	31	Equal/Unequal
1	30	Greater Than/Equal
0	29	Limits
3	28	Fixed PT Overflow
8	27	Task use of 05-4
9	26	Remove Interrupt Lockouts
10	25	Int/Task Accumulators
11	24	Int/Task Base Regs.
N/A	22-23	01 = Indirection 10 = Optional Indirection 11 = Character Addressing
N/A	21	Repeat Mode
15	20	Halfword Indicator
N/A	0-19	PAR

ACTIVE STATUS REGISTER

TABLE V-1

The remaining bits in the ASR are not used in this Emulation, but if needed could be mapped into any unused location in memory below address 1024.

The D-Machine's A1 register is used to maintain a copy of the instruction currently being executed. All other registers are free for computational use. The D-Machine's

General Condition Bit One (GC1) is reserved for communicating with the Loader. General Condition Bit Two (GC2) is used to indicate that an AN/UYK-7 instruction is being executed under a special condition; such as, repeat mode or indirection.

The Emulator consists of an Opcode/Sub-opcode Directory, an instruction fetch routine (IFETCH), global subroutines and opcode execution subroutines. When the Emulator receives a signal from the Loader (GC1) to commence excution, it immediately passes control to IFETCH, which reads the PAR and places the first instruction in A1. The opcode portion of the instruction is isolated and used as a pointer into the Opcode Directory. Control is then passed to the opcode execution subroutine designed to handle that particular instruction. The opcode subroutines may stand alone and perform their functions independently; or they may call several global subroutines used to consolidate code for multiple opcodes. Upon completion of its execution, the opcode subroutine passs control back to IFETCH which then fetches the next instruction.

In the case of halfword instructions, control is passed to HALFFETCH which serves the same purpose as IFETCH, except for 16 bit instructions. Every attempt is made within the opcode subroutines to do in-line coding versus subroutine calls. This decreases execution time considerably; although, it greatly increases program size. During the initial programming phase the Interpreters were limited to a

4K micromemory due to the dynamic overlay routine (SMX) being positioned at 4k. As a result, excessive subroutine calls were necessary in order to keep the program size below 4K. This software deficiency has since been corrected by Burrough's and a full 8K micromemory is now available. These subroutine calls could now be removed and the subroutines moved back in-line.

Emulation of Central Processor instructions in the AN/UYK-7 repertoire was fairly straightforward. Most instructions were implemented with an average of 15 microinstructions. This generally included: isolating the various fields, forming the effective operand (Y) address, fetching the operands from main memory, performing the instruction function, and writing the result back to main memory, commonly referred to as S-Memory. Because of the parallelism of the D-Machine's nanoinstructions several of these functions were performed simultaneously. Many AN/UYK-7 instructions have sub-opcodes, each of which perform a different operation on the same operands. In such cases, the opcode subroutine centralized code by performing the field isolation and operand fetch functions prior to calling the sub-opcode subroutine. Either the operand itself or its address was passed in a register to the subroutine, which then performed its function and wrote the results back to S-Memory.

The difficulties arose in the AN/UYK-7 Emulation when the repeat and indirection modes of operation were implemented. The Repeat Instruction required that the next

sequential instruction be executed the number of times contained in the Index Register B(7), or until specified alternate conditions were met. This required that flags be set indicating that the repeat mode had been initiated and that special termination conditions were to be tested. Condition testing took place after each execution of the repeated instruction and prior to return to IFETCH for the next instruction. If the conditions were not satisfied IFETCH was bypassed and the same instruction was reexecuted after first decrementing B(7), and then indexing the operand address by the displacement value contained in the repeat instruction. If the conditions were met the normal IFETCH mode was executed and the repeat mode was terminated. Most instructions required checking the condition of a specified accumulator register to signal completion of the repeat mode; however, compare instructions used the results of their comparison to signal completion. Repeated replace instructions used a different Y-operand calculation for their store phase then for their fetch phase. Instructions that were not designated as being repeatable were executed once and the repeat mode was terminated. A copy of the Repeat Instruction was saved in S-Memory at address 0030; a copy of the instruction being repeated was saved at address 0031; and the Y-operand address used for storing during a repeated replace instruction was saved at 0033. These addresses are given in octal and are normally unused in the AN/UYK-7.

The indirect modes of operation possible in the AN/UYK-7 were extremely difficult to emulate. There are four modes of indirection possible: Normal Indirection, Character Addressing, Sequential Character Addressing, and Optional Indirection. Not all instructions are indirectable and some instructions are indirectable but not character addressable. To make this determination a table was created in the Emulator which would return a value based on the instruction opcode. This value indicates whether an instruction is repeatable, indirectable, or character addressable. All modes of indirection are initially signalled by the i-field of the instruction being a one. This indicates that the Y-operand points to an Indirect Control Word (ICW). The ICW, which is presented in Figure III-1, is then fetched and its bits 30-31 are examined to determine the indirection mode. In all cases, if the i-field of the ICW is set, the Y-operand again points to a new ICW. The Y-operand is calculated in accordance with the indirect mode. Operand fetching will continue in this manner, cascading through memory, until an ICW is found without its i-field set. At this point the lower 20 bits of the ICW replaces the lower 20 bits of the original instruction.

Normal indirection is indicated by a binary 10 in bits 31-30 of the ICW. Y-operand address calculation is performed as Y=y+B(b)+S(s). Once the final ICW is determined then normal execution of the instruction resumes.

Single Character Addressing is indicated by a binary 01 in bits 31-30 of the ICW. Y-operand address calculation is performed as Y=y+B(b)+S(s); however, in this mode two additional fields (P,W) of the ICW are meaningful. The P-field indicates the least significant bit position of the character to be fetched from the operand and the W-field indicates the number of bits in the character. When the character is fetched - it is right justified; operated on in accordance with the instruction; and then (depending on the instruction) ORed back into its original position in the Y-operand.

Sequential Character Addressing indicated by a binary 11 in bits 31-30 of the ICW, operates in exactly the same manner as Single Character Addressing except in the case where the store cycle is required. Before ORing the resulting character back into the Y-operand the P and W-fields are compared. If P minus W is positive then the difference replaces P and the character is stored accordingly. If the difference is negative then 32 minus W replaces P and the "sy" field of the ICW is incremented by 1 and replaced in memory for the next execution.

Optional Indirection is indicated by a binary 00 in bits 31-30 of the ICW. In this mode bit 29 of the ICW is also significant and modifies the Y-operand address calculation. If bit 29 is 0 then Y=sy+S(b). If bit 29 is 1 then Y=sy+B(b)+(S), where S is designated by bits 17-19 of B(b). Once the first operand is fetched then normal instruction execution occurs.

A large portion of the Emulation code was devoted to handling the four modes of indirection and its ad hoc address calculations. General Condition Bit 2 (GC2) was used to flag the indirect and repeat modes of operation. Address 0032, octal, in S-Memory was reserved for saving the current ICW being used during Character Addressing; address 0033, octal, was reserved for the Y-operand address in the case of optional indirection; address 0034, octal, was reserved for the address of the current ICW which could be updated during sequential character addressing. These addresses are part of the Control Memory Registers and are not normally used in the AN/UYK-7.

Although the IOC has not been emulated, Input/Output functions were implemented which make the systems' peripherals accessible to the user. The AN/UYK-7 Initiate I/O instruction was implemented in the Emulator with the exception that the a-field, which is normally used as a channel designator, actually designates the desired peripheral. Based on the contents of the a-field, a function code is formulated and passed to the IOC portion of the Loader. The Emulator then enters a neutral state awaiting an Interrupt from the Loader that the I/O has been performed. The Y-operand address of the I/O instruction fetched by the Emulator points to the address of the buffer, rather than to a buffer control word as in the AN/UYK-7. It was felt that

implementation of I/O in this manner would remain virtually transparent to the user, and would facilitate full emulation of IOC functions at a later date.

Of the 132 instructions in the AN/UYK-7 Repertoire, 111 have been fully implemented and tested. One instruction (Return Jump) has been implemented but not tested. Twenty instructions remain to be both written and tested. All unwritten instructions were assigned space in the Opcode Directory and, if used by the programmer, will cause an error message to be printed to the effect that the instruction has not been yet implemented. The implementation of these instructions involves developing the required microcode and inserting it into the space already allocated in the Opcode Subroutine Library. The unimplemented instructions include all Floating Point Operators, Scale Factor, and Interrupt Handlers.

The Repeat Instruction and its associated mode of operation have been implemented and tested with the exception of the repeat replace instructions, which have been written but not yet tested. Indirect addressing has been fully implemented but has not been tested. Input/Output functions of the IOC have been implemented and tested allowing the user to read a card, print a buffer, or read a disk sector. The disk write function has been designed into the Emulator but not implemented for fear of destroying resident disk material during execution. This function should not be fully employed until either a monitor system is developed which

would preclude write access to assigned disk sectors, or the assumption is made that the entire disk is available to the user as a scratch pad.

C. REGISTER MAPPING

Because the D-Machine has few registers available to the user, the authors chose to map all of the AN/UYK-7 registers into main memory. In addition, it was necessary to create several special locations to hold temporary results or conditions. In order to facilitate this mapping, all memory locations below 1024 decimal are reserved; thus, all user instructions and memory references were offset by this amount by the Loader. Since this Emulation will be in a monoprogramming environment, all Base Registers were initialized in increments of 8K with the first Base Register set to 1024. This allowed user access to approximately 63K of main memory, with each Base Register referencing an 8K page of memory. The Control Memory Register (CMR) allocations and additional temporary mappings (indicated by an *), are presented as part of Figure V-2. The registers marked "*" are not used in the AN/UYK-7, and were thus assigned special purposes in the Emulator. Additional unused register areas are available and labelled "unused" in Figure V-2. These register areas could be employed for the same purposes if needed in the course of expansion to a full emulation.

OCTAL ADDRESS	DECIMAL ADDRESS	DESCRIPTION
0-7	0-7	Task A-Reg. (0-7)
10	8	Unused
11-17	9-15	Task Index B-Reg. (1-7)
20-27	16-23	Task Base S-Reg. (0-7)
*30	24	Repeat Instruction Temp.
*31	25	Current Instruction being Repeated
*32	26	Current Indirect Control Word (ICW)
*33	27	Y-operand for Indirection
*34	28	ICW Address in Memory
*35	29	Program Address Register
36-57	30-47	Unused
60-67	48-55	Interrupt Breakpoint Registers (0-7)
70-77	56-63	Active Status Words (0-7)
100-107	64-71	Int. A-Registers (0-7)
110	72	CP Monitor Clock
111-117	73-79	Int. Index B-Reg. (1-7)
120-127	80-87	Int. Base S-Registers (0-7)
*130	88	Switch Values Temporary
131-137	89-95	Unused
140-157	96-103	Designator Storage Words (DSW)
160-167	104-111	Storage Protection Registers (SPR)
170-177	112-119	Segment Identification Registers (SIR)
*200	120	Location Counter
*201	121	Real Time Clock
*202-246	122-166	Disk Input Buffer
*247-323	167-211	Disk Output Buffer
*324-357	212-231	Card Input Buffer
360-365	232-237	Unused
*366-426	238-270	Printer I/O Buffer
427-437	271-279	Unused
*440-1777	280-1023	Error Routines
2000-77775	1024-65533	User Program Area
*77775-77776	65534-65535	Mailbox

*-Temporary Location Established By Programmers Convention

REGISTER - MEMORY MAPPING

Figure V-2

D. TIMING

Although no formal timing survey was conducted, an analysis of the Emulation itself provided an indicator which was used in developing a comparison to real-time. For this analysis it was assumed that each instruction was executed using direct addressing and not in repeat mode. The IFETCH routine used 16 microseconds which was an automatic overhead acquired by all instructions. The operand fetch and store routines (YFETCH, YSTORE and Y2FETCH) added an additional 33, 30, or 14 microseconds, respectively. Thus any instruction which referenced a Y-operand incurred an additional overhead of 14 microseconds at a minimum. Adding to this overhead, the execution time of the basic opcode itself yielded the time estimated for typical instructions given in Table V-2.

INSTRUCTION TIME	OPERANDS	EMULATION TIME	AN/UYK-7 TIME
Load	Register,Y	60	1.5
Add	Register,Y	63	1.5
Add	Register	36	1.0
Divide	Register,Y	600	15

TIME ESTIMATES

TABLE V-2

Using these figures it was estimated that this Emulation ran on the order of 40 times real-time. Two steps could be taken that would considerably improve this time. First, move all out-of-line subroutine calls in-line in the opcode subroutines. Second, enhance the Interpreter as discussed under Instruction Timing in Chapter III. It has been estimated that this same Emulation could run at approximately 5 times real-time on an enhanced D-Machine. This time compares favorably with a Simulation of the AN/UYK-7 written in PL-360 for the IBM 360, which runs on the order of 1000 times real-time. As discussed in Chapter II, it would be difficult to approach real-time in emulating the AN/UYK-7 because of the parallelism of its fetch and field isolation operations. The addition of a Field Select Unit (FSU) and additional internal temporary registers would be required before a real-time emulation could be realized.

VI. SUMMARY

The authors' conclusions and recommendations resulting from any research project are a fundamental part of the formal presentation and are therefore included in this thesis. In addition, the authors felt that the reader, who might be contemplating an emulation project, would be interested in the problem areas encountered during the pursuit of this thesis. Accordingly this chapter is divided into two sections - Problems and Conclusions.

A. PROBLEMS

During the course of this emulation numerous software and hardware problems were encountered which had a considerable effect on progress and the amount accomplished in the allocated time. Software problems stemmed principally from poor documentation of the D-Machine, the AN/UYK-7, their associated languages and internal configuration. The software problems with the D-Machine were generally resolved through experimentation on the machine or through phone calls to Burrough's Advanced Development Organization (ADO). The lack of thorough AN/UYK-7 documentation proved to be a more severe handicap since no test base for experimentation was accessible at the Naval Postgraduate School. Such a medium is absolutely mandatory when working in the emulation

environment. The programmer must be able to determine the machine reaction to unorthodox instruction configurations. Some degree of assistance was provided by the Fleet Combat Director Systems Support Activity (FCDSSA), San Diego; however, in some cases the result was a "best guess" as to what the AN/UYK-7 response would be.

Hardware problems are to be expected during the course of any major project and particularly when a new machine is involved. This Emulation was the first project to be undertaken on the Naval Postgraduate School's D-Machine since the machine was installed. The estimated mean-time-between-failures (MTBF) experienced by the authors was less than five hours. Problems ranged from loose circuit boards and dirty contacts, to disk read/write head misalignment. Unfortunately, the time to repair was aggravated by the fact that the school has no technicians trained on the D-Machine, and the authors were inexperienced in its maintenance and operating procedures. Without the assistance provided by Burrough's ADO the project could not have been seen through to fruition. The ADO provided the school with engineering assistance on three separate occasions, the last of which uncovered a major design problem. The machine has since been modified and the system is presently very reliable.

In addition to these two major problem areas progress was inhibited by the following:

1. Lack of Software Utilities - Since the D-Machine was a new installation the school has not had the opportunity to generate the numerous desirable software utilties. In an effort to eliminate the inherent slowness of a "Card Only" system, the authors have undertaken the implementation of the supervisor's console as an adjoint to this thesis. The authors are also implementing a Text Editor which will allow for on-line program modification, and will replace the present card input Line Editor.

2. Microprogramming is not a new concept at the Naval Postgraduate School; however, actual experience was lacking. The Computer Science Department has played a major role in implementation of the D-Machine and the required expertise is rapidly being developed.

3. The learning curve for undertaking an emulation is tremendous. The target machine, the host machine, their respective programming languages and internal configurations must be learned and understood in minute detail.

4. The target machine for this Emulation, the AN/UYK-7, is an extremely sophisticated and difficult to understand computer. The central processor instruction set is fairly straightforward and easy to implement; however, the numerous and differing modes of operation severely complicated the Emulation.

B. CONCLUSIONS

The authors feel that they have successfully achieved their goal of demonstrating the feasibility of emulating the AN/UYK-7 on the Burrough's D-Machine. The design presented in this thesis is a workable design as evidenced by execution of AN/UYK-7 programs successfully and by execution in

less than 40 times the actual execution time of the AN/UYK-7. Better than 90% of the AN/UYK-7 Instruction Repertoire has been implemented and tested, thus providing a sound foundation for a full emulation. The design allows for expansion to a full emulation which would include the IOC instruction repertoire and Floating Point without major modification. Further, the authors have concluded that by modification of the design to place out-of-line subroutine calls in-line and enhancement of the D-Machine, the Emulation would execute within 5 times the actual speed of the AN/UYK-7.

During the course of this project the authors had an opportunity to visit the Naval Surface Weapons Center (NSWC), Dahlgren, Virginia. Emulation in general was discussed with the programming staff, who had just completed emulation of the Trident Computer on the Nanodata QM-1, and who were contemplating emulating the AN/UYK-7 on the QM-2. Based on these discussions, the experience gained in the course of this thesis, and presupposing the present level of knowledge of the authors; it is estimated that a complete emulation of the AN/UYK-7, its IOC and all Interrupt functions would require an additional 2 man years of programming effort. This estimate includes a reasonably sophisticated degree of testing, and assumes full accessibility of the D-Machine and an AN/UYK-7 test base.

VII. RECOMMENDATIONS

The authors feel that this thesis has provided them with a learning experience previously unparalleled in either of their lives, since it has entailed detailed and independent learning of emulation, microprogramming (TRANSLANG), the Burrough's D-Machine and the AN/UYK-7. In addition, the authors became part time technicians in an effort to keep the D-Machine running. It is felt that the same educational horizons await any student considering a similar project; however, in an effort to keep others from having to "reinvent the wheel", the authors have included much of what was learned about these subjects in this thesis. Hopefully, by building on these experiences, others will be able to begin at a much higher level and consequently accomplish more. The following recommendations are submitted for consideration by anyone inclined to undertake a thesis involving emulation.

1. Do no try to emulate a machine that uses hardware interrupts on an Interpreter that does not have interrupts.

2. Pick a target machine that is well defined and documented. Computers are designed to perform a specific set of functions; the results of unorthodox use are not always known, but must be considered by the emulator. The availability of a target machine for testing to determine such results is mandatory.

3. Ensure that the host Interpreter is well documented and has a reasonable set of system utilities; such as, an editor, debugger, simulator, file manager, operating system and compiler.

4. Ensure the host Interpreter will receive hardware support if problems arise.

5. Ensure publications are available covering utilites, hardware, languages and target/host machine capabilities and operation.

6. Be prepared to work independently and have your questions answered through your own experimentation.

In addition to these recommedations, the authors would like to propose several thesis topics applicable to the D-Machine.

1. Several projects could be undertaken as a direct follow on to this thesis:

   a. An Operating System, File Manager, and Assembler could be written in AN/UYK-7 machine language utilizing the existing Emulator.

   b. The present Emulation could be enhanced to include Floating Point and any unimplemented instructions, including incorporation of the IOC functions and allowing for synchronous I/O.

   c. A timing survey could be conducted comparing the existing Emulation with AN/UYK-7 benchmarks.

2. Several thesis projects are possible which entail enhancing the capabilities of the existing D-Machine Installation. These include writing the interface to the PDP-11/45, and a resident Text Editor.

The stand alone environment of the D-Machine lends itself to experimentation with the design of operating systems and file managers.

In summary, the authors would like to state that they found the D-Machine an outstanding learning device in spite of its many hardware problems. The authors found the ability to emulate existing computers exciting, and found the realization that they could build a working model of any computer they could design, exhilarating.

APPENDIX A. USER'S MANUAL

A. D-MACHINE

When the D-Machine was installed at the Naval Postgraduate School, it was not accompanied by any form of documentation for its operation, preventive maintenance, diagnostics or utility programs. For the most part this information has been derived through experimentation and frequent calls to Burrough's ADO. Some of the more pertinent information is included in this User's Manual to enable the user to load and run a program such as the AN/UYK-7 Emulation. Unless specified otherwise all addresses given in this discussion are in hexadecimal.

1. System Initializing

The machine is secured every night and must therefore, be reinitialized when powered up each morning.

a. Ensure circuit breakers 2, 7 and 10 are on.

b. Push the "ON" button on the disc and all three Interpreters.

c. With desired discs in the drives, push the two "RUN" buttons on the disk unit.

d. Turn on the card reader and printer.

e. Place the IOP in normal vice single step mode via the switch inside the cabinet on the right.

f. Push the "LOAD" button on the IOP; then push "CLEAR".

g. Place the cards marked IOP-Loader into the card reader. Cards should be read and the IOP should stop at an MPAD of 00FA. If not, repeat steps e through g.

h. Push the "LOAD" button on IOP to return the IOP to normal.

i. Perform steps e to h for each of the other two Interpreters using the cards marked STK-loader. They should halt at a MPAD of 004A.

k. Push "CLEAR" on each Intrepreter. They should bootstrap themselves through the IOP and halt at either 0542 or 051A. (An Interpreter must be at 0542 to run a program and only one Interpreter can be at 0542 at a time.) Clear the IOP to force an Interpreter to switch from 051A to 0542.

The D-Machine now thinks it is a Burrough's B6700 single-stack machine. Any utility can be run using the proper control cards as documented in the computer lab. The more common utilities are Sunrise (updates the day, date and time; should be run first thing every morning), Dir-List (prints out the directory; system or user), Compile (compiles ALGOL programs) and TRANSLANG (assembles microprograms). By use of the proper control cards, the user can also obtain a source listing of his program as it is being compiled or assembled. Actually the default is a printout every time and the user must insert a "$-list clist" control card to eliminate the listing.

To make the computer act as some machine other than the B6700, use the control card "?SMLOAD filename". The "?" must be in column one and the rest of the card is free-format. This loads the microprogram "filename" (which must be an assembled TRANSLANG program) into micro-memory overlaying the stack machine and then transfers control to that program starting at address 0000. Thus to load the AN/UYK-7, execute "?SMLOAD UYK7-LOADER" on one Interpreter and "?SMLOAD ANUYK7-EMULATOR" on the other. Refer to the appropriate sections of this chapter for further instructions on the execution of these programs.

2. Diagnostics

The programs used for diagnosing hardware problems are written in TRANSLANG and maintained as object modules on cards. The appropriate program is loaded in the same fashion as the bootstrap loader and terminates at a significant address. When "cleared", it should again terminate at an address which indicates success or failure. The operator must determine which from the program listing, as no messages are printed. There are no diagnostics written for the disk, printer or card reader. At the present time a CRT terminal is hooked up to the IOP; however, there is no software support for it. An interface is being written into the operating system and eventually all commands to and from the operator will be via the CRT. The card reader and line printer will then be strictly data I/O ports.

3. Debugging

While executing any TRANSLANG program, the user can monitor the program by use of the nixie light display panel and the function switch assigned to each processor. The lights indicate hex numbers. By selecting the appropriate function the user can observe the current nanoinstruction being executed; the address of the current microinstruction (MPAD); the memory address last written to or read from (BMAR); and the MIR register (MIR and EXT). The MIR function displays the lower 16 bits of the MIR register while the EXT function displays the upper 16 bits. These functions are presently the only means of debugging. For example, if the user were unsure of the information being read from S-memory, he could insert two additional instructions in the code: an instruction to place the data into the MIR register and a "WAIT" instruction. The CPU halts at the "WAIT" instruction and the user can then examine the information in MIR to ascertain its validity. The user may check the MPAD to determine if the address read was actually the address intended. To continue the program the user must press the force-step button inside the side panel. If the user wished to step through a series of instructions following the "WAIT", he could go to single-step, and while holding the force-step button, press the single-step button to override the "WAIT". After passing the "WAIT", single-step to execute one instruction at a time. In this manner the user can debug several instructions one at a time.

B. AN/UYK-7

This section of the User's Manual discusses how the user can reconfigure the D-Machine into an AN/UYK-7.

First, ensure that the disk packs which are mounted are the AN/UYK-7 System Packs and that both interpreters are loaded with an ALGOL Stack Machine.

Second, insert an "?SMLOAD UYK7-LOADER" card in the card reader. This will cause the Loader's microcode to overlay the stack machine's microcode and stop at address "0000".

Third, insert an "?SMLOAD UYK7-EMULATOR" card in the reader. This will cause the Emulator's microcode to overlay the alternate stack machine and stop at address "00A0". The AN/UYK-7 is now ready to accept and execute programs.

Fourth, place the AN/UYK-7 source programs in the reader and force step the Loader and the Emulator, respectively. The Loader will read one card at a time, decode it and place the 32-bit resulting instruction in memory for subsequent execution by the Emulator.

The last instruction of all user programs should be a "HALT", which will cause the Emulator to halt and the Loader to be reinitialized for the next job. Force stepping the Emulator after the halt will cause the next job to be read and executed. Use of this convention permits batch processing of programs by the AN/UYK-7. The program deck organization is discussed in the next section. Individual programs are separated by "N" cards.

1. Loader

a. Macros

This section describes how a program must be keypunched for input to the Loader. All Macro control characters and formats are listed in Figure A-1. Any character typed in column one, which is not listed, will cause an "Illegal Instruction" message and the card will be ignored. It is noteworthy that all functions are optional with the exception of the "N", which must terminate the program deck. All addresses are in octal and considered absolute as far as the user is concerned; however, they are offset by 1024 by the Loader. Macros and instructions may be interspersed as desired.

COL. 1	COL. 4-8	FUNCTION
"A" Register 0-7		Cols. 9-19 contain the octal value to be inserted in the listed A-Reg.
"I" Register 1-7		Cols. 9-19 contain the octal value to be inserted in the listed B-Reg.
"D" Octal Address		Cols. 9-16 contain the decimal value to be inserted at the address listed or in-line if the address=0000.
"R" Octal Address		Cols. 9-19 contain the decimal number of words to be saved at the address listed or in-line if the address=0000.
"W" Octal Address		Cols. 9-80 contain the character string to be inserted at the address specified. The string terminates before col. 80 if a single quote (') is encountered.
"S" Switch Value		Cols. 6-8 contain an octal number whose bit pattern selects which of eight AN/UYK-7 switches the user desires set.
"L" Octal Address		Causes Base S-Registers to be set at 8K pages starting at 1024, and the program to be loaded at Address.
"O" Octal Address		Causes change in the program counter in order that subsequent instructions may be loaded at Address.
"N" Octal Address		Terminates loading of program and initializes PAR to Address for the Emulator.

LOADER MACRO DEFINITIONS

FIGURE A-1

b. Instruction Preparation

For inputting instructions to the Loader three basic formats are used which correspond roughly to the five formats used by the AN/UYK-7. The Loader formats are presented in Figure A-2 under their respective AN/UYK-7 Format Headings. All fields require a right justified octal representation. The "i" field should always be either a one or a zero since it represents a single bit. The combined "F3,K" field of Format III instructions requires that K always be zero; therefore, valid inputs are (0,2,4,6) which correspond to an F3 of (0,1,2,3). Columns 1-3 must always be left blank. Columns 15-80 may be used for programmer's comments. Each instruction is decoded into one 32-bit word and assigned to the next sequential address in memory. However, if the instruction is a half-word instruction then it will be stored in the lower half of the previous word if that word also contained a half-word instruction, else it will be stored in the upper half of the next sequential address. An instruction's sequential address assignment can only be changed via the "O" or "L" macros, which must precede the instruction. A sample program is presented in Appendix D.

FORMAT I

COL 1-3	COL 4-5	COL 6	COL 7	COL 8	COL 9	COL 10-14
BLANK	OPCODE	A-REG	K	B	I	Y-OPERAND

FORMAT II

COL 1-3	COL 4-5	COL 6	COL 7	COL 8	COL 9	COL 10-14
BLANK	OPCODE	A-REG	F2	B	I	Y-OPERAND

FORMAT III

COL 1-3	COL 4-5	COL 6	COL 7	COL 8	COL 9	COL 10-14
BLANK	OPCODE	A-REG	F3,K	B	I	Y-OPERAND

FORMAT IV-A

COL 1-3	COL 4-5	COL 6	COL 7	COL 8	COL 9
BLANK	OPCODE	A-REG	F4	B	i

FORMAT IV-B

COL 1-3	COL 4-5	COL 6	COL 7-9
BLANK	OPCODE	A-REG	SHIFT DESIGNATOR

INSTRUCTION FORMATTING

FIGURE A-2

c. TRANSLANG

Several modifications have been made to the D-Machine and the microprogramming language, TRANSLANG, since the publication of reference 9 in 1970. Unfortunately, an addendum to reference 9 has not been published and these powerful modifications remain undocumented. Because the authors took advantage of these capabilities in writing this Emulation the changes are documented here to provide the user a ready reference. They include: additional logical operators, program address modifiers, and an additional Y-select input register (BMAR).

(1) Logic Operators. The additional logic operators are listed in Table A-1 along with the code generated by the Microtranslator when they are encountered. In the Table, A1 and B are assumed as the X/Y input operands respectively. In addition, listed under the columns labelled TRUE/FALSE, are the tests generated by the Microtranslator when a TRUE/FALSE conditional test is encountered subsequent to the logical operation.

OP	MEANING	CODE GENERATED	TRUE TEST	FALSE TEST
LSS	Less Than	A1-B	NOT AOV	AOV
LEQ	Less Than or Equal	A1-B-1	NOT AOV	AOV
EQL	Equal	A1 EQV B	ABT	NOT ABT
NEQ	Not Equal	A1 EQV B	NOT ABT	ABT
GEQ	Greater Than or Equal To	A1-B	AOV	NOT AOV
GTR	Greater Than	A1-B-1	AOV	NOT AOV

LOGICAL OPERATORS

TABLE A-1

The programmer is cautioned that when the Microtranslator encounters the keywords TRUE/FALSE it backtracks to the last logical operator to determine which test to generate. Thus a TRUE/FALSE conditional test which is arrived at through branching may not have generated the exact code the programmer anticipated. These operators were intended to be used in a sequential fashion as follows:

```
A1 GTR B
If TRUE Then (operation)
```

The Microtranslator generates:

```
A1-B-1
If AOV Then (operation)
```

The following illustrates circumstances in which the Microtranslator does not generate the code expected by the programmer:

```
A1 LSS B
If LC1 Then A1 GTR B;Step Else Skip
If TRUE Then (operation)
```

Generates the code:

```
A1-B
If LC1 Then A1-B-1;Step Else Skip
If AOV Then (operation)
```

In this situation "If AOV" is always generated when the TRUE is encountered; although, obviously if LC1 was not set then the programmer desired to test for "Less Than".

(2) Program Address Modifiers. Two powerful new instructions were added to the D-Machine which allow direct transfer of program control. These are the two Type II instructions: "Label-1=MPCR" and "Label-1=CPCR". The first causes a direct change in the program sequence by loading MPCR with the new Label. The second can be classified as a return jump since it causes a transfer of AMPCR to MPCR, and MPCR+1 to AMPCR; thereby setting up a program jump to a subroutine, and saving the return address. This code is identical to:

```
Label-1=AMPCR
CALL
```

The only difference between them is that the CPCR requires only one instruction for execution, while the CALL requires two instructions.

(3) BMAR. The BMAR register is formed by the concatenation of BR1 or BR2 (16 bits) and MAR (8 bits). It can be used as a Y-Input of 24 bits to the Adder. The actual BR register selected is the one most recently referenced

in an external operation. The user is cautioned in the use of BMAR because of its odd size of 24 bits. The unwary programmer could unintentionally allow extraneous bits in the most significant byte when BMAR is used for temporary storage of 16-bit addresses. That is, the assignment "A1=MAR" is considered to effect a transfer of 16 bits, while in fact 24 bits are transferred. A subsequent statement of "BMAR=A1" could permit extraneous bits to be introduced in the third least significant byte, if the programmer had made the assumption that only 16 bits were involved and automatic masking had taken place. The programmer can force selection of BR1 or BR2 by issuing an ASR or ASE, respectively, prior to a BMAR reference. These are considered external function NOOPS in the Naval Postgraduate School Configuration; however, they may not appear concurrently with a BEX statement.

The remainder of the TRANSLANG statements are reasonably straightforward once the programmer understands the instruction timing idiosyncracies. Reference 9 provides excellent guidance for the novice microprogrammer.

APPENDIX B. EMULATOR PROGRAM LISTING

This appendix provides a copy of the Microtranslator output listing of the Emulator. A copy of the source program is maintained on cards and also on the Burrough's D-Machine disk. The object module produced by the Microtranslator is maintained on disk. Each line of the program is divided into four sections. The leftmost grouping consists of the hexadecimal address to which the microinstruction was assigned by the Microtranslator during assembly. This is followed by the 56-bit microinstruction which was generated. The middle group consists of the programmers' source coding which was input to the Microtranslator. The rightmost group consists of the sequence numbers which were assigned to each input card as it was inserted into the source file on the disk by the Interpreter's resident line-editting program (CARD-LIST).

```
                        $MERGE ANUYK7-EMULATOR
                        PROGRAM ANUYK7-OBJECT                                                   0001000C
                        ABASE=0                         %ADDRESS OF A-REG(C)                    0C020C00 D
                        PAR=29                          %TEMP. STORAGE LOCATION OF PAR          0C030C00 D
                        BBASE=8                         %ADDRESS OF B-REG(0)                    0C040C00 D
                        SBASE=16                        %ADDRESS OF S-REG(0)                    0C050C00 D
                              S6=22                     %ADDRESS OF BASE REG 6                  0006CC00 D
                        ZERO=0.  ONE=1                                                          0007CC00 D
                        CARDBUF=212                     %ADDRESS OF PERMANENT CARD BUFFER       0C080C00 D
                        PRINTBUF=238                    %ADDRESS OF PERMANENT PRINT BUFFER      0C090C00 D
                        RINST=24                        %TEMP. LOCATION OF REPEAT INSTR.        0C1CCC00 D
                        CINST=25                        %TEMP. LOCATION OF INSTR. REPEATED      0C110000 D
                        IAW=26                          %TEMP. LOCATION OF CURRENT IAW          0C12CC00 D
                        YADDR=27                        %CONTAINS PRECALCULATED Y-ADDRESS       0013CC00 D
                        IAWADDR=28                      %ADDRESS OF CURRENT IAW                 0C140C00 D
                        SWITCH=88                       %ADDRESS OF SWITCH VALUES READ-IN       0015CC00 D
                                                                                                00160C00 D
CC00  0000 0003 0C30 0040       START-1 = MPCR                                                  0017CC00 D
0C01  0010 0000 0030 00C0       ILLEGAL-1=AMPCR         %INVALID OPR 00                         0018CC00 D
0C02  0020 0000 0C30 C0C0       OPR01-1=AMPCR           %INCLUSIVE OR                           0C19CC00 D
0C03  0030 00C0 0030 C0CC       OPR02-1=AMPCR           %COUNT ONES                             0C1ACC00 D
0C04  0040 0000 0C30 C0C0       OPR03-1=AMPCR           %REPLACE INCLUSIVE OR                   0020CC00 D
CC05  005C 0000 0C30 00C0       ILLEGAL-1=AMPCR         %INVALID OPR 04                         00210C00 D
CC06  0060 0000 0030 C0C0       OPR05-1=AMPCR           %DOUBLE LOAD A                          C022CC00 D
0C07  0070 00C3 0C30 00C0       OPR06-1=AMPCR           %FP ADD                                 0023CC00 D
0C08  008C 0003 0030 00C0       OPR07-1=AMPCR           %ENTER EXEC STATE                       0C240C00 D
0C09  0090 0000 0C30 00C0       OPR10-1=AMPCR           %LOAD A                                 0025CC00 D
CC0A  00A0 0003 0030 C0C0       OPR11-1=AMPCR           %LOAD A AND INDEX                       0C260C0C D
0C0B  00B0 0000 0C30 00CC       OPR12-1=AMPCR           %LOAD DIFFERENCE                        0C27CC00 D
0C0C  00C0 0000 0030 C0C0       OPR13-1=AMPCR           %SUBTRACT A                             0028CC00 D
0C0D  00D0 0003 0030 C0CC       OPR14-1=AMPCR           %ADD A                                  0029CC00 D
0C0E  00E0 C003 0030 00CC       OPR15-1=AMPCR           %LOAD SUM                               0C30CC00 D
0C0F  00F0 C0C0 0030 00C0       OPR16-1=AMPCR           %LOAD NEGATIVE                          0C31CC00 D
0C10  010C 0000 0C30 00C0       OPR17-1=AMPCR           %LOAD MAGNITUDE                         0C32CC00 D
0C11  0110 0000 0030 00C0       OPR20-1=AMPCR           %LOAD B                                 0C330C00 D
0C12  0120 0003 0030 C0C0       OPR21-1=AMPCR           %ADD B                                  0034CC00 D
0C13  0130 0000 0030 00C0       OPR22-1=AMPCR           %SUBTRACT B                             0035CC00 D
0C14  014C 0000 0030 00C0       OPR23-1=AMPCR           %STORE B                                00360C00 D
0C15  0150 0000 0030 C0C0       OPR24-1=AMPCR           %STORE A                                00370CC0 D
CC16  0160 C000 0C30 C0C0       OPR25-1=AMPCR           %STORE A AND INDEX B                    00380C00 D
CC17  017C 0003 0030 00C0       OPR26-1=AMPCR           %STORE NEGATIVE                         0C390C00 D
0C18  0180 C003 0030 00C0       OPR27-1=AMPCR           %STORE MAGNITUDE                        0C400C00 D
0C19  0190 0CC0 0C30 00C0       ILLEGAL-1=AMPCR         %INVALID OPR 3C                         00410C00 D
0C1A  01AC 00C0 003C C0C0       ILLEGAL-1=AMPCR         %INVALID OPR 31                         0C420C00 D
0C1B  01BC 0000 0030 C0C0       OPR32-1=AMPCR           %CLEAR BIT                              00430C00 D
0C1C  01CC 0000 0000 C0CC       OPR33-1=AMPCR           %SET BIT                                0C440C00 D
0C1D  01DC 0000 0030 C0C0       OPR34-1=AMPCR           %REPALCE ADD                            0C450C00 D
0C1E  01E0 0000 0C30 00C0       OPR35-1=AMPCR           %REPLACE INCREMENT                      0C460C00 D
0C1F  01F0 00C3 0030 00C0       OPR36-1=AMPCR           %REPLACE SUBTRACT                       0C470C00 D
0C20  020C 0003 0C30 00C0       OPR37-1=AMPCR           %REPLACE DECREMENT                      00480C00 D
0C21  0210 0003 0030 00C0       OPR40-1=AMPCR           %MULTIPLY A                             0C490C00 D
0C22  022C 0000 0000 00C0       OPR41-1=AMPCR           %DIVIDE A                               0050CC00 D
0C23  0230 0003 0C30 C0CC       OPR42-1=AMPCR           %COMPARE BIT TO ZERO                    0C510C00 D
0C24  024C 0000 0C30 00C0       OPR43-1=AMPCR           %COMPARE INDEX INCREMENT                00520C00 D
0C25  025C 0003 0030 C0C0       OPR44-1=AMPCR           %COMPARE                                0053CC00 D
0C26  0260 00C3 0030 C0C0       OPR45-1=AMPCR           %COMPARE LIMITS                         0C54CC00 D
0C27  0270 0003 0C30 00C0       OPR46-1=AMPCR           %COMPARE MASKED                         0C55CC00 D
0C28  028C 0000 0C30 00C0       OPR47-1=AMPCR           %COMPARE GATED                          0C560C00 D
0C29  0290 0003 0030 00C0       OPR50-1=AMPCR           %JUMP ON EVEN PARITY                    0057CC0C D
```

```
0C2A  02AC  0000 0000 00C0          OPR51-1=AMPCR          %JUMP A POSITIVE                          00580C00 D
0C2B  02B0  0000 0000 00C0          OPR52-1=AMPCR          %LOAD B AND JUMP                          00590C00 D
0C2C  02C0  0000 0000 C0C0          OPR53-1=AMPCR          %JUMP ON NO OVERFLOW                      005C0C00 D
0C2D  02D0  0000 0000 00C0          OPR54-1=AMPCR          %LOAD CMR TASK                            005D0C00 D
0C2E  02E0  0000 0000 00C0          OPR55-1=AMPCR          %LOAD CMR INCREMENT                       005E0C00 D
0C2F  02F0  0000 0000 C0C0          OPR56-1=AMPCR          %STORE CMR TASK                           005F0C00 D
0C30  0300  0000 0000 00C0          OPR57-1=AMPCR          %STORE CMR INTERRUPT                      00600C00 D
0C31  0310  0000 0000 00C0          OPR60-1=AMPCR          %STORE CMR IN A                           00610C00 D
0C32  0320  0000 0000 C0C0          OPR61-1=AMPCR          %LOAD CMR FROM A                          00620C00 D
0C33  0330  0000 0000 00C0          OPR62-1=AMPCR          %SHIFT LEFT CIRCULARLY                    00630C00 D
0C34  0340  0000 0000 00C0          OPR63-1=AMPCR          %SHIFT LEFT CIRCULARLY DOUBLE             00640C00 D
0C35  0350  0000 0000 00C0          OPR64-1=AMPCR          %SHIFT RIGHT FILL ZEROS                   00650C00 D
0C36  0360  0000 0000 00C0          OPR65-1=AMPCR          %SHIFT RIGHT DOUBLE FILL ZEROS            00660C00 D
0C37  0370  0000 0000 C0C0          OPR66-1=AMPCR          %SHIFT RIGHT FILL SIGN                    00670C00 D
0C38  0380  0000 0000 00C0          OPR67-1=AMPCR          %SHIFT RIGHT DOUBLE FILL SIGN             00680C00 D
0C39  0390  0000 0000 00C0          OPR70-1=AMPCR          %SCALE FACTOR                             00690C00 D
0C3A  03A0  0000 0000 00C0          OPR71-1=AMPCR          %LOGICAL SUMM                             006A0000 D
0C3B  03B0  0000 0000 0040          ILLEGAL4-1=MPCR        %PRINT MESSAGE & RETURN TO HALFFETCH      006B0C00 D
0C3C  03C0  0000 0000 0040          ILLEGAL4-1=MPCR        %PRINT MESSAGE & RETURN TO HALFFETCH      006C0C00 D
0C3D  03D0  0000 0000 00C0          OPR74-1=AMPCR          %MULTIPLY REGISTER                        006D0C00 D
0C3E  03E0  0000 0000 0040          ILLEGAL4-1=MPCR        %PRINT MESSAGE & RETURN TO HALFFETCH      006E0C00 D
0C3F  03F0  0000 0000 0040          ILLEGAL4-1=MPCR        %PRINT MESSAGE & RETURN TO HALFFETCH      006F0C00 D
0C40  0400  0000 0000 C0C0          OPR77-1=AMPCR          %STORE IOC MONITOR CLOCK IN A             00700C00 D
                                  OPR01XX:                                                          0C710C00 D
0C41  0410  0000 0000 C0C0          OPR01X0-1=AMPCR                                                  00720C00 D
0C42  0420  0000 0000 00C0          OPR01X1-1=AMPCR                                                  0C730C00 D
0C43  0430  0000 0000 C0C0          OPR01X2-1=AMPCR                                                  0C740C00 D
0C44  0440  0000 0000 C0C0          OPR01X3-1=AMPCR                                                  0C750C00 D
0C45  0450  0000 0000 00C0          OPR01X4-1=AMPCR                                                  0C760C00 D
0C46  0460  0000 0000 00C0          OPR01X5-1=AMPCR                                                  0C770C00 D
0C47  0470  0000 0000 00C0          OPR01X6-1=AMPCR                                                  00780C00 D
0C48  0480  0000 0000 00C0          OPR01X7-1=AMPCR                                                  0C790C00 D
                                  OPR02XX:                                                          007A0C00 D
0C49  0490  0000 0000 00C0          OPR02X0-1=AMPCR                                                  007B0C00 D
0C4A  04A0  0000 0000 00C0          ILLEGAL-1=AMPCR                                                  0C7C0C00 D
0C4B  04B0  0000 0000 C0C0          OPR02X2-1=AMPCR                                                  007D0C00 D
0C4C  04C0  0000 0000 C0C0          OPR02X3-1=AMPCR                                                  007E0C00 D
0C4D  04D0  0000 0000 C0C0          OPR02X4-1=AMPCR                                                  0C7F0C00 D
0C4E  04E0  0000 0000 00C0          OPR02X5-1=AMPCR                                                  0C800C00 D
0C4F  04F0  0000 0000 00C0          OPR02X6-1=AMPCR                                                  00810C00 D
0C50  0500  0000 0000 00C0          OPR02X7-1=AMPCR                                                  0C820C00 D
                                  OPR03XX:  %USES OPR01XX ROUTINES DUE TO DUPLICATION OF CODE       00830C00 D
                                  OPR05XX:                                                          00840C00 D
0C51  0510  0000 0000 C0C0          OPR05X0-1=AMPCR                                                  00850C00 D
0C52  0520  0000 0000 00C0          OPR05X1-1=AMPCR                                                  0C860C00 D
0C53  0530  0000 0000 C0C0          OPR05X2-1=AMPCR                                                  0C870C00 D
0C54  0540  0000 0000 C0C0          OPR05X3-1=AMPCR                                                  0C880C00 D
0C55  0550  0000 0000 C0C0          OPR05X4-1=AMPCR                                                  0C890C00 D
0C56  0560  0000 0000 00C0          ILLEGAL-1=AMPCR                                                  008A0C00 D
0C57  0570  0000 0000 00C0          ILLEGAL-1=AMPCR                                                  008B0C00 D
0C58  0580  0000 0000 00C0          ILLEGAL-1=AMPCR                                                  0C8C0C00 D
                                  OPR06XX:                                                          008D0C00 D
0C59  0590  0000 0000 00C0          OPR06X0-1=AMPCR                                                  008E0C00 D
0C5A  05A0  0000 0000 C0C0          OPR06X1-1=AMPCR                                                  008F0C00 D
0C5B  05B0  0000 0000 00C0          OPR06X2-1=AMPCR                                                  0C900C00 D
0C5C  05C0  0000 0000 C0C0          OPR06X3-1=AMPCR                                                  0C910C00 D
0C5D  05D0  0000 0000 00C0          OPR06X4-1=AMPCR                                                  0C920C00 D
0C5E  05E0  0000 0000 C0C0          OPR06X5-1=AMPCR                                                  0C930C00 D
0C5F  05F0  0000 0000 00C0          OPR06X6-1=AMPCR                                                  0C940C00 D
0C60  0600  0000 0000 00C0          OPR06X7-1=AMPCR                                                  0C950C00 D
```

```
                                        OPR07XX:                                                
```

```
0096  096C 0000 0030 00C0          OPR74X6-1=AMPCR                                                    0178CC00 0
0097  097C 0000 0000 00CC          OPR74X7-1=AMPCR                                                    0179CC00 0
                                 OPR77XX:                                                           0180CC00 0
0098  0980 00C0 0030 00C0          OPR77X0-1=AMPCR                                                    0181CC0C 0
0099  0990 0003 0030 00C0          OPR77X1-1=AMPCR                                                    0182CC00 0
009A  09A0 0000 0000 0040          ILLEGAL4-1=MPCR          %PRINT MESSAGE & RETURN TO HALFFETCH      0183CC00 0
009B  09B0 0000 0030 0040          ILLEGAL4-1=MPCR          %PRINT MESSAGE & RETURN TO HALFFETCH      0184CC00 0
009C  09C0 0000 003C 00CC          OPR77X4-1=AMPCR                                                    0185CC00 0
009D  09D0 0000 0030 00C0          OPR77X5-1=AMPCR                                                    0186CC00 0
009E  09E0 0000 0030 00C0          OPR77X6-1=AMPCR                                                    0187CC00 0
009F  09F0 00C0 0030 0040          ILLEGAL4-1=MPCR          %PRINT MESSAGE & RETURN TO HALFFETCH      0188CC00 0
                                 %                                                                  0189CC00 0
                                 %  START BY TRYING TO SET GC1 WHICH MEANS THE                      0190CC00 0
                                 %  LOADER HAS LOADED THE OBJECT CODE AND                           0191CC00 0
                                 %  PRESET THE REGISTERS                                            0192CC00 0
                                 %  IN GENERAL A2 CONTAINS ASR/PAR, A1 CONTAINS CURRENT INSTRUCTION 0193CC00 0
00A0  4800 0000 0020 C0F0        START:  0 = BR1; WAIT                                              0194CC00 0
00A1  0988 00C0 0000 00FC          SET GC1; WHEN GC1 THEN STEP                                      0195CC00 0
00A2  0A00 00C0 0C30 C060          GETPAR-1 = CPCR                                                  0196CC00 0
00A3  48C9 00C1 0B30 00F0          1 L=B;RESET GC2          %ADD 1024 TO STARTING ADDRESS           0197CC00 0
00A4  A00C 00C0 0030 0020          COMP 10=SAR                                                      0198CC0C 0
00A5  4809 CC40 2030 00F0          A2+B=A2                  %A2 = STARTING ADDRESS OFFSET 1024      0199CC00 0
                                 IFETCH:                                                            0200CC00 0
00A6  0A10 00C0 0C30 0060          PUTPAR-1 = CPCR                                                  0201CC00 0
00A7  B80B 00C0 0C30 00F0          IF IRQ THEN STEP ELSE SKIP                                       0202CC00 0
00A8  0A20 0000 0030 0060          IODUMPREG-1 = CPCR                                               0203CC00 0
00A9  180B 0000 0C30 00F0          IF GC2 THEN STEP ELSE SKIP                                       0204CC0C 0
00AA  0A30 00C0 0C30 C060          GC2CHECK-1 = CPCR                                                0205CC00 0
                                 IFETCH1:                                                           0206CC00 0
00AB  4809 C0C3 001C 00FC          A2 = MAR2                                                        0207CC0C 0
00AC  A809 0000 0030 0CF0          MR2; IF ROC              % READ NEXT INST.                       0208CC0C 0
00A0  AC08 C0C0 2C00 00F0          WHEN RDC THEN A2 + 1 = A2,BEX    % INC PAR                       0209CC00 0
                                 IFETCHREMOTE:   %ENTRY FROM EXECUTE REMOTE OPR02X2                 0210CC00 0
00AE  4809 0C40 4000 00F0          B = A1                   % A1 => CURRENT INST.                   0211CC00 0
00AF  4809 0C40 8B00 00F0          B R = B                                                          0212CC0C 0
00B0  0000 00C0 0000 0020          16 = SAR                                                         0213CC00 0
00B1  4809 0C40 0030 00F0          B                                                                0214CC00 0
00B2  580B 0000 CC30 C0F0          IF LST THEN STEP ELSE SKIP                                       0215CC00 0
00B3  0A40 0000 0C30 C060          INDIRECT-1 = CPCR        % INDIRECTION                           0216CC00 0
                                 IFETCH2:                                                           0217CC00 0
00B4  4809 A0C0 8B00 00F0          A1 R = B                                                         0218CCC0 0
00B5  1070 0003 0000 00A0          7 = LIT; 17 = SAR                                                0219CC00 0
00B6  4809 2C56 0B30 00FC          LIT AND B = B            % E-FIELD                               0220CC00 0
00B7  4809 A0C0 8040 00F0          A1 R = AMPCR             % OPCODE                                0221CC00 0
00B8  2080 0000 0C00 C0B0          BBASE = LIT; 26 = SAR                                            0222CC00 0
00B9  4809 0000 0C30 00F0          STEP                                                             0223CC00 0
00BA  2824 2C40 100C 00F0          LIT + B = MAR,A3; EXEC; IF LC1    % B(B) ADDR =>MAR,A3           0224CC00 0
00BB  382D 0000 0000 00F0          JUMP; IF LC2             % JUMP TO OPCODE                        0225CC00 0
                                 %                                                                  0226CC00 0
                                 %                                                                  0227CC00 0
                                 % ********************     SUBR FOR MEMORY FETCHES      ************0228CC00 0
                                 HALFFETCH:      %FETCH THE NEXT HALFWORD INSTRUCTION               0229CC00 0
00BC  9809 0000 003C 18FC          MW1;IF SAI               %WRITE RESULTS OF PREVIOUS HALFWORD     0230CC0C 0
                                   %SOME INSTRUCTIONS ENTER HERE AT HALFFETCH+1                     0231CC0C 0
00B0  4809 A0C1 C000 C0F0          A1 C=A1                  %FETCH LOWER HALFWORD                   0232CC00 0
00BE  0000 0000 0000 0020          16=SAR                   %SWITCH UPPER AND LOWER WORDS           0233CC00 0
00BF  4809 A0C0 900C 00F0          A1 R=A3                  %ISOLATE & SAVE OPCODE                  0234CCC0 0
00C0  200C 00C0 0C30 0030          26=SAR                                                           0235CC00 0
00C1  4809 C001 8B30 40FC          A2 C=B,CSAR                                                      0236CC00 0
00C2  8000 0000 0000 C020          20=SAR                   %GET HALFWORD FLAG TO LSB               0237CC00 0
```

```
00C3  4809 0C40 003C 00F0          B                          %TEST BIT SET=>PREVIOUS HALFWORD     02380C00 D
00C4  5C08 0C01 A000 00F0          IF LST THEN BITO C=A2; STEP ELSE SKIP  %SET BIT IF CLEARED      02390C00 D
00C5  0A50 0000 0000 C040          IFETCH-1=MPCR                                                   02400000 D
00C6  3809 CCC1 A030 00FC          BIT1 C=A2;IF LC2           %CLEAR REPEAT BIT IF SET;GET NEXT    02410C00 D
00C7  B019 0000 0000 00F0          IF NOT IRO THEN SKIP                                            02420C00 D
00C8  0A5C 0003 0030 0060          PUTPAR-1=CPCR                                                   02430000 D
00C9  B019 0000 0030 00F0          IF NOT IRO THEN SKIP                                            02440C00 D
00CA  0A6C 0000 0C30 0060          IODUMPREG-1=CPCR                                                02450C00 D
00CB  4809 EC00 0C40 00F0          A3=AMPCR                                                        02460C00 D
00CC  4809 0000 0030 00F0          STEP                                                            02470C00 D
00CD  2824 E012 0C00 00F0          A3 EQL 0;EXEC;IF LC1       %IF LOWER OPC=C DON;T EXECUTE        02480C00 D
00CE  6029 0000 0030 C0F0          IF FALSE THEN JUMP                                              02490C00 D
00CF  0BC0 00C3 0C30 0040          HALFFETCH=MPCR                                                  02500000 D
                                %                                                                  02510C00 D
                                YFETCH:    %MAR=B(B) ADDRESS;RETURN MAR2=Y-ADDR;MIR=Y(K)           02520C00 D
00D0  A809 A000 9030 00F0          A1 R=A3;IF RDC                                                  02530000 D
00D1  8070 0003 0030 00A0          7=LIT;20=SAR               %ISOLATE K FIELD                     02540C00 D
00D2  A809 E156 1C35 08F0          A3 AND LIT=CTR,A3;MR1;IF RDC %SAVE K IN CTR;READ B(B)           02550C00 D
00D3  1019 0000 0030 C0F0          IF NOT GC2 THEN SKIP       %GC2 SET IF SPECIAL CONDITIONS       02560C00 D
00D4  0A70 00C0 0C00 0040          YABNORMAL-1=MPCR           %CHECK FOR SPECIAL CONDITIONS        02570C00 D
00D5  AC08 00C0 0C30 00F0          WHEN RDC THEN BEX          %B=B(B)                              02580C00 D
00D6  4809 E002 0000 C0F0       YREPFETCH:  NOT A3                                                 02590C00 D
00D7  6C19 0C40 0030 C0F0          IF ABT THEN B=MIR;SKIP     %CHECK K=0                           02600C00 D
00D8  0A8C 0000 0C30 C040          KNOTZERO-1=MPCR            %%JUMP HERE IF KNOT = ZERO           02610C00 D
00D9  4809 0018 8B30 00F0       KZERO:  B111 R=B              %ARRIVE HERE FROM ABNORMAL NOT CA    02620000 D
00DA  000C 0000 0000 0020          16=SAR                                                          02630C00 D
00DB  4809 A0C1 1030 00F0          A1 L=A3                    %CHECK SIGN OF SY-FIELD              02640C00 D
00DC  4809 EC00 0000 00F0          A3                         %A3= SY SHIFTED LEFT TO MSB          02650C00 D
00DD  4C19 EC50 9000 00F0          IF MST THEN A3 OR B C=A3,BMI;SKIP  %EXTEND NEGATIVE SIGN        02660C00 D
00DE  4809 E000 9000 00F0          A3 R = A3,BMI              %EXTEND POSITIVE SIGN                02670C00 D
00DF  482D EC40 1C30 00F0          A3+B=A3,MIR;JUMP           %Y=SY(SE)+B(B);GO TO OPCODE          02680C00 D
00E0  4809 A003 9000 00F0       KNOTZERO:  A1 R=A3            %ISOLATE S-FIELD                     02690C00 D
00E1  9070 0000 0000 0090          7=LIT;13=SAR                                                    02700C00 D
00E2  4809 E156 103C C0F0          A3 AND LIT=A3                                                   02710C00 D
00E3  4809 E140 0C3C 00F0          A3+LIT=MAR                 %MAR=ADDRESS OF S(S)                 02720000 D
00E4  01C0 00C0 0030 C0E0          SBASE=LIT                                                       02730C00 D
00E5  A809 A0C1 1030 48FC          A1 L=A3,CSAR;MR1;IF RDC    %READ S(S)                           02740C00 D
00E6  4809 E0C0 9030 00F0          A3 R=A3                    %A3=Y-FIELD ISOLATED RIGHT           02750C00 D
00E7  AC08 EC40 1C00 C0F0          WHEN RDC THEN A3+B=A3,BEX  %B=S(S); A3=Y+B(B)                   02760C00 D
00E8  4809 EC40 001C 00F0          A3+B=MAR2                                                       02770C00 D
                                YFETCH1:   %RETURN HERE FROM Y-ABNORMAL UNLESS K=0 OR CHAR ADDR.   02780C00 D
00E9  4809 0640 0C3C 24F0          AMPCR=MIR;ASE              %SELECT BR2=K; SAVE RETURN           02790C00 D
00EA  4809 0BC2 9000 00F0          NOT CTR R=A3               %RESTORE K IN MSB INPUT FROM CTR     02800C00 D
00EB  000C 0000 0000 0030          24=SAR                     %SHIFT CTR TO LSB                    02810C00 D
00EC  4809 E640 009C 00F0          A3+AMPCR=AMPCR             %SET UP K-HANDLER ROUTINES           02820C00 D
00ED  0A9C 0000 0030 00CC          KTABLE-1=AMPCR                                                  02830C00 D
00EE  A809 0000 0030 CCF0          MR2;IF RDC                 %READ Y VALUE                        02840C00 D
00EF  4824 0000 0030 00F0          EXEC                       %EXECUTE K-HANDLER ROUTINE           02850C00 D
00F0  4836 00C0 0000 C0F0          CALL                                                            02860C00 D
00F1  4809 0C40 0040 C0FC          B = AMPCR                                                       02870C00 D
00F2  4809 0C00 0030 00FC          STEP                                                            02880C00 D
00F3  482D 00C0 003C C0F0          JUMP                                                            02890C00 D
00F4  480C 0000 0C00 00F0       KTABLE: WAIT                                                       02900C00 D
00F5  0AAC 0000 0CC0 C0C0          KR1-1=AMPCR. KR2-1=AMPCR. KR3-1=AMPCR. KR4-1=AMPCR              02910C00 D
00F6  0ABC 00C3 0C3C 00C0
00F7  0AC0 00C0 0030 00C0
00F8  0A00 0000 0C00 00C0
00F9  0AE0 00C0 0030 C0C0          KR5-1=AMPCR. KR6-1=AMPCR. KR7-1=AMPCR                           02920000 D
00FA  0AF0 0000 0000 C0C0
00FB  0B0C 0000 0C00 00C0
```

```
00FC  000C 0000 0030 0020      KR1:    16 = SAR                                                    0293CC00 D
00FD  AC08 00C0 0C30 C0FC              WHEN RDC THEN BEX           % READ (Y)                      0294CC00 D
00FE  4809 0C41 1030 C0F0              B L = A3                                                    0295CC00 D
00FF  4809 E0C0 9D30 C0F0              A3 R = A3,MIR,BMI           % (Y) 0-15 = MIR                0296CC00 D
0100  4C0D 0C40 0C30 00F0              IF MST THEN B=MIR; STEP ELSE JUMP                           0297CC00 D
0101  4809 0019 0B30 00F0              B111 L = B                  % 1:S IN BITS 16-31             0298CC00 D
0102  482D EC5C 1D30 00F0              A3 OR B = A3,MIR,BMI;JUMP   % SIGN EXTENDED                 0299CC00 D
0103  000C 00C0 0000 0020      KR2:    16 = SAR                                                    0300CC00 D
0104  AC08 0000 0C30 C0F0              WHEN RDC THEN BEX           % READ (Y)                      0301CC00 D
0105  4809 0C40 9D30 00F0              B R = A3,MIR,BMI            % (Y) 31-16 = A3                0302CC00 D
0106  4C0D 0C40 0030 00F0              IF MST THEN B=MIR;STEP ELSE JUMP                            0303CC00 D
0107  4809 0019 0B00 C0F0              B111 L = B                                                  0304CC00 D
0108  482D EC5C 1D30 C0F0              A3 OR B = A3,MIR,BMI; JUMP %EXTEND SIGN                     0305CC00 D
0109  AC08 00C0 0C30 00F0      KR3:    WHEN RDC THEN BEX           % READ (Y)                      0306CC00 D
010A  482D 0C40 1D30 00F0              B = A3,MIR,BMI;JUMP                                         0307CC00 D
010B  0FFC 00C3 0C30 C0E0      KR4:    255 = LIT                                                   0308CC00 D
010C  AC08 0000 0C30 C0F0              WHEN RDC THEN BEX           % READ (Y)                      0309CC00 D
010D  482D 2C56 1D3C 00F0              LIT AND B = A3,MIR,BMI;JUMP                                 0310CC00 D
010E  0FF0 0000 003C C090      KR5:    255 = LIT; 8 = SAR                                          0311CC0C D
010F  AC08 00C0 0C30 00FC              WHEN RDC THEN BEX           % READ (Y)                      0312CC00 D
0110  4809 0C40 8B30 00F0              B R = B                                                     0313CC00 D
0111  482D 2C56 1D30 C0F0              LIT AND B = A3,MIR,BMI;JUMP     % (Y) 15-8 = MIR            0314CC00 D
0112  0FF0 0000 0030 00A0      KR6:    255 = LIT; 16 = SAR                                         0315CC00 D
0113  AC08 0000 0C30 C0F0              WHEN RDC THEN BEX           % READ ( Y)                     0316CC00 D
0114  4809 0C40 8B3C 00F0              B R = B                                                     0317CC00 D
0115  482D 2C56 1D30 00F0              LIT AND B = A3,MIR,BMI; JUMP    % (Y) 23-16 = MIR           0318CC00 D
0116  0FF0 0000 0C30 C0B0      KR7:    255 = LIT; 24 = SAR                                         0319CC00 D
0117  AC08 0000 0C30 00F0              WHEN RDC THEN BEX           %READ (Y)                       0320CC00 D
0118  4809 0C40 8B3C 00F0              B R = B                                                     0321CC00 D
0119  482D 2C56 1D30 C0F0              LIT AND B = A3,MIR,BMI; JUMP    % (Y) 31-24 = MIR           0322CC00 D
                               %                                                                  0323CC00 D
                               YABNORMAL: %CALLED FROM Y-FETCH WHEN GC2 SET MAR=B-FIELD;A3,BR2=K-FIELD 0324CC00 D
011A  AC08 E0C0 0B35 C0F0              WHEN RDC THEN A3=B,CTR                                      0325CC00 D
011B  4809 C0C0 9000 00F0              A2 R=A3                                                     0326CC0C D
011C  9070 0000 0030 00A0              21=SAR;7=LIT                                                0327CC00 D
011D  4809 E156 9C30 00F0              A3 AND LIT R=A3             %A3=SPEC.COND.BITS              0328CC00 D
011E  11BC 0000 0030 0080              1=SAR;YADDR=LIT                                             0329CC00 D
011F  5C0B 0C40 1C00 C0F0              IF LST THEN B=A3,BEX;STEP ELSE SKIP                         0330CC00 D
0120  0D5C 00C3 0030 0040              YREPFETCH-1=MPCR                                            0331CC00 D
0121  4809 00C0 0038 C0F0              LMAR                        %READ PRECALC. Y-ADDRESS        0332CC00 D
0122  A809 0000 0C0C 08F0              MR1; IF RDC                                                 0333CC00 D
0123  AC08 E152 0C00 00F0              WHEN RDC THEN A3 EQL LIT,BEX  %TEST=3=>CHAR.ADDR;E=Y-ADDR   0334CC00 D
0124  0030 0000 0C30 00B0              3=LIT;24=SAR                                                0335CC00 D
0125  6419 0C43 001C C0F0              IF FALSE THEN B=MAR2;SKIP   %NOT CA SO SETUP Y-ADDRESS      0336CC00 D
0126  0B1C 00C3 0000 0040              CHARFETCH-1=MPCR            %MUST HAVE BEEN 3=>CHAR.ADDR.   0337CC00 D
0127  0E80 00C3 0000 C040              YFETCH1-1=MPCR                                              0338CC00 D
                               %                                                                  0339CC00 D
                               Y2FETCH:                            %ISOLATE (Y) AND LEAVE IN B     0340CC00 D
0128  180B 0000 0030 C0F0              IF GC2 THEN STEP ELSE SKIP                                  0341CC00 D
0129  0B2C 0000 0030 0040              Y2ABNORMAL-1=MPCR                                           0342CC00 D
012A  4809 A0C1 1000 40F0      Y2REPABN:  A1 L=A3,CSAR                                            0343CC00 D
012B  907C 0000 0030 0090              COMP 19=SAR;7=LIT                                           0344CC00 D
012C  A809 E0C0 9000 48F0              A3 R=A3,CSAR;MR1;IF RDC     %ISOLATED THE Y-FIELD;READ B(B) 0345CC00 D
012D  AC08 0000 0C30 00F0              WHEN RDC THEN BEX           %B=B(B)                         0346CC00 D
012E  4809 EC40 1030 C0F0              A3+B=A3                                                     0347CC00 D
012F  4809 A0C0 8B0C 00F0              A1 R=B                      %Y+B(B)=A3                      0348CC00 D
0130  4809 2C56 0B3C C0F0              LIT AND B=B                 %ISOLATE S-FIELD                0349CC00 D
0131  4809 2C40 0C0C 00F0              LIT+B=MAR                                                   0350CC00 D
0132  010C 00C0 0C30 00E0              SBASE=LIT                                                   0351CC00 D
0133  A809 0000 0000 C8F0              MR1;IF RDC                                                  0352CC00 D
```

```
0134  AC08 0000 0C00 00FD          WHEN ROC THEN BEX                                                    0353CC00 D
0135  4809 EC40 0C1C 00FD          A3+B=MAR2                    %B=Y-ADDR IF SPECIAL NOT CA             0354CC00 D
                                Y2FETCH1:  %NORMAL RETURN FROM ~2ABNORMAL                             0355DC00 D
0136  A809 0000 0000 CCF0          MR2;IF ROC                                                           0356DC00 D
0137  AC28 0000 0C30 00FD          WHEN ROC THEN BEX;JUMP       %(Y) NOW IN B REGISTER                  0357CC00 D
                                Y2ABNORMAL:  %CALLED FROM Y2 WHEN GC2 SET; MAR=B-REG                  0358CC00 D
0138  4809 CC00 9030 00F0          A2 R=A3                                                              0359CC00 D
0139  9000 0000 0000 0D20          21=SAR                                                               0360CC00 D
013A  4809 E156 9030 00F0          A3 AND LIT R=A3                                                      0361CC00 D
013B  1070 00C0 0C30 C08D          7=LIT;1=SAR                                                          0362CCD0 D
013C  5419 0003 DC08 00F0          IF NOT LST THEN LMAR;SKIP                                            0363CC00 D
013D  129C 00C0 D030 0040          Y2REPABN-1=MPCR                                                      0364CC00 D
013E  01BC 0003 0030 C0E0          YADDR=LIT                                                            0365CC00 D
013F  A809 0000 0030 08F0          MR1;IF ROC                                                           0366CC00 D
0140  AC08 E152 0C0C 00F0          WHEN ROC THEN A3 EQL LIT,BEX %TEST FOR CAR,ADDR.;E=B(B)              0367CC00 D
0141  0C30 0000 0C0D DDE0          3=LIT                        %CHECK FOR CHARACTER ADDR.              0368CC00 D
0142  640B 0C43 001C 0DF0          IF FALSE THEN B=MAR2;STEP ELSE SKIP %IF NOT CA THEN SETUP Y-ADDR     0369CC00 D
0143  1350 0000 0030 0040          Y2FETCH1-1=MPCR              %NOT AHAR ADDRESSING                    0370CC00 D
0144  0B3C 0000 0C30 004D          CHARFETCH-1=MPCR             %CHARACTER ADDRESSING                   0371CC00 D
                                %                                                                     0372CC00 D
                                CHARFETCH:   %FETCH CHARACTER ADDRESSED BY Y=(B)                      0373CC00 D
0145  4809 0C40 009C 00F0          B=MAR2,MIR                   %READ (Y);SAVE Y IN MIR                 0374CC00 D
0146  A809 0000 DC30 0CF0          MR2;IF ROC                                                           0375CC00 D
0147  AC08 2000 DC3C C0FC          WHEN ROC THEN LIT=MAR,BEX  %B=(Y);MAR=ICW ADDRESS TEMP.              0376CC00 D
0148  01AC 0003 0030 00E0          IAW=LIT                                                              0377CC00 D
0149  A809 0C40 1000 08F0          B=A3;MR1;IF ROC              %A3=(Y)                                 0378CC00 D
014A  AC08 00C0 0C3C C0F0          WHEN ROC THEN BEX            %B=ICW                                  0379CC00 D
014B  4809 0C40 8B00 C0F0          B R=B                        %B==W/P                                 0380CC00 D
014C  825C 00C0 0030 00A0          20=SAR;37=LIT                %MASK FOR W&P FIELDS                    0381CC00 D
014D  4809 2C56 D030 80F0          LIT AND B=SAR                %SAVE P-FIELD                           0382CC00 D
014E  4809 E0C0 9C30 0DF0          A3 R=A3                      %RIGHT JUSTIFY CHAR BY P                0383CC00 D
014F  4809 0C43 8B0C 0DF0          B R=B                        %B=0C/W IN LSB                          0384CC00 D
0150  9000 0000 DC30 0000          5=SAR                                                                0385CC00 D
0151  4809 2C56 0000 80F0          LIT AND B=SAR                %W VALUE => SAR                         0386CC00 D
0152  4809 00C0 0030 40F0          CSAR                         %COMPLEMENT SAR = NOT W                 0387CC0C D
0153  4809 0013 8B00 00F0          B111 R=B                     %MASK FOR W BITS                        0388CC00 D
0154  4809 EC56 1D90 00F0          A3 AND B=A3,MIR,BM1          %A3=DATA MASKED;B=Y ADDRESS             0389CC00 D
0155  482D 0C43 001C C0F0          B=MAR2;JUMP                  %B=Y ADDRESS                            039DCC00 D
                                %RETURN TO CALLING OP YFETCH OR Y2FETCH;MIR=CHAR IN LSB               0391CC00 D
                                %                                                                     0392CCC0 D
                                %                                                                     0393CCC0 D
                                % ########################    SUBR. FOR MEMORY STORES      ############0394CC00 D
                                YSTORE:   %CALLED FROM OPCODES VALUE TO STORE IS IN MIR                0395CC0C D
                                %MAR1 IS SETUP TO READ B(B)                                           0396DC00 D
0156  4809 A000 9C00 00FD          A1 R=A3                      %ISOLATE B-FIELD                        0397CC00 D
0157  1070 00C0 0000 00A0          17=SAR;7=LIT                                                         0398CC00 D
0158  4809 E156 1030 00F0          A3 AND LIT=A3                                                        0399CCCC D
0159  4809 E140 003C C0F0          A3+LIT=MAR                                                           040CCC0C D
015A  008C 00C0 0030 00E0          BBASE=LIT                                                            0401CC00 D
015B  A809 A000 8B00 08F0          A1 R=B;MR1;IF ROC            %READ B(B)                              0402CC0C D
015C  8070 0003 DC30 0DA0          7=LIT;20=SAR                                                         0403CC00 D
015D  4809 2C56 1030 00FD          LIT AND B=A3                                                         0404CC00 D
015E  0B40 D0C0 0C0C 00C0          KSTABLE-1=AMPCR                                                      0405CC0C D
015F  AC08 E640 0C40 00F0          WHEN ROC THEN A3+AMPCR=AMPCR,BEX                                     0406CC00 D
0160  1419 A001 1000 40F0          IF NOT GC2 THEN A1 L=A3,CSAR;SKIP                                    0407CC00 D
0161  0B50 0000 0000 CD40          YSABNORMAL-1=MPCR                                                    0408CC00 D
0162  9070 0000 0000 0090          COMP 19=SAR;7=LIT                                                    0409CCCC D
                                YSTORE2:   %RETURN HERE FROM ABNORMAL IF NOT REPLACE ABNORMAL         0410CC00 D
0163  4809 E0C0 9030 00F0          A3 R=A3                      %ISOLATE Y-FIELD                        0411CC00 D
0164  4809 EC40 1030 40F0          A3+B=A3,CSAR                 %Y+B(B)=A3                              0412CC00 D
```

```
0165  4809 A000 8B30 00F0          A1 R=B                                                                  0413CC00 D
0166  4809 2C56 0B90 00F0          LIT AND B=B                 %ISOLATES S-FIELD                           0414CC00 D
0167  4809 2C40 0C3C C0F0          LIT+B=MAR                                                               0415CC00 D
0168  010C 0D00 0C00 C0A0          16=SAR; SBASE=LIT           %SAR SETUP FOR USE BY KS ROUTINES           0416CC00 D
                                YSTORE1:  %RETURN FERE FROM ALL INDIR EXCEPT CA AND REPLACE REPEAT        0417CC00 D
0169  A809 0003 0030 C8FC          MR1; IF RDC                                                             0418CC00 D
016A  AC20 0000 0C00 00FC          WHEN RDC THEN BEX;EXEC                                                  0419CC00 D
016B  4809 EC43 0C1C C0F0          A3+B=MAR2                   %MAR2=Y+B(B)+S(S)                           0420CC00 D
016C  A82D 0000 0C0C 0CF0          JUMP;MR2;IF RDC                                                         0421CC00 D
                                YSABNORMAL:  %CALLED FROM YSTORE; AMPCR=K-HANDLER;B=B(B)                   0422CC00 D
016D  4809 C000 9C30 C0F0          A2 R=A3                     %ISOLATE SPEC COND BITS                     0423CC00 D
016E  9070 0003 0000 C0A0          21=SAR; 7=LIT                                                           0424CC00 D
016F  4809 E156 9000 00F0          A3 AND LIT R=A3                                                         0425CC00 D
0170  1000 0000 0C30 0000          1=SAR                                                                   0426CC00 D
0171  5419 E152 0000 00FC          IF NOT LST THEN A3 EOL LIT;SKIP                                         0427CC00 D
0172  0B6C 0CC0 0000 0040          YSREP-1=MPCR                %MUST BE REPEAT STORE                       0428CC00 D
0173  0030 0000 003C 00E0          3=LIT                       %CODE FOR CHAR ADDR.                        0429CC00 D
0174  6819 00C0 003C 00F0          IF TRUE THEN SKIP           %TEST FOR CA                                0430CC00 D
0175  0B70 0000 0000 0040          YSABN-1=MPCR                %NOT CHAR ADDRESSING                        0431CC00 D
0176  0B8C 0000 0000 0060          CHARSTORE-1=CPCR            %CA STORE CHAR AND RETURN                   0432CC0C D
0177  0A50 0003 0C30 0040          IFETCH-1=MPCR                                                           0433CC00 D
                                YSABN;  %REPEAT/OPT. INDIR./INDIR. NOT CHAR ADDR.                         0434CC00 D
0178  4809 03C0 1038 00F0          0=A3,LMAR                   %READ PRECALCULATED ADDRESS OF Y            0435CC00 D
0179  01BC 0000 003C C0E0          YADDR=LIT                   %ADDRESS OF PRECALC. Y-ADDR.                0436CC00 D
017A  16B0 0000 0030 0040          YSTORE1-1=MPCR              %LET YSTORE READ/WRITE USING K              0437CC00 D
017B  3419 A3C1 1000 40F0      YSREP:  IF NOT LC2 THEN A1 L=A3,CSAR;SKIP                                  0438CC00 D
017C  177C 0000 0000 C040          YSABN-1=MPCR                %REPLACE THEN USE PRECALC ADDR OF Y         0439CC00 D
017D  9070 00C0 0000 0090          COMP 19=SAR;7=LIT                                                       0440CC00 D
017E  162C 0000 0000 C040          YSTORE2-1=MPCR              %NOT REPLACE REPEAT CALC ADDRESS            0441CC00 D
                                %                                                                         0442CC00 D
                                CHARSTORE:  %CALLED FROM YSTORE IF CHAR ADDR.;MASK WORD IN MIR BY W BITS0443CC00 D
                                        %SHIFT BACK TO P POSIT. AND OUTPUT TO Y. IF SINGLE CA TO Y ADDR.0444CC00 D
                                        %IN (19) ELSE TO Y OR Y+1 AT (19) DEPENDING ON LC1=P-W LT 0 = P 0445CC00 D
                                        %OR P-W GE 0 = P;UPDATE P AND STORE AT ADDRESS (2C).             0446CC00 D
017F  4809 0000 0038 C0F0          LMAR                        %READ ICW                                   0447CC00 D
0180  01AC 00C0 003C C0E0          IAW=LIT                                                                 0448CC00 D
0181  A809 0000 003C 08F0          MR1; IF RDC                                                             0449CC00 D
0182  AC08 0000 0C00 00F0          WHEN RDC THEN BEX           %B=ICW                                      0450CC0C D
0183  4809 0C40 903C 00F0          B R=A3                      %ISOLATE BITS 31 & 30                       0451CC00 D
0184  A030 0000 003C 00B0          30=SAR; 3=LIT                                                           0452CC00 D
0185  4809 E152 0000 00F0          A3 EOL LIT                  %SEQUENTIAL CA CHECK                        0453CC00 D
0186  6419 0C40 903C 00F0          IF FALSE THEN B R=A3; SKIP %B=ICW;A3=OC/W/P                             0454CC00 D
0187  0B90 0000 0C3C 0040          STORESEO-1=MPCR                                                         0455CC00 D
0188  81FC 00C0 003C 00A0          20=SAR;31=LIT               %MASK FOR P&W FIELDS OF IAW                 0456CC00 D
0189  4809 E156 403C C0F0          A3 AND LIT =A1              %A1 NOW = P VALUE                           0457CC00 D
018A  4809 E0C0 8B08 C0F0          A3 R=B,LMAR                 %LIT=ADDRESS OF Y-ADDRESS;A3=OC/W           0458CC00 D
018B  91B0 0000 003C 00B0          YADDR=LIT;5=SAR                                                         0459CC00 D
018C  A809 2C56 0000 88F0          LIT AND B=SAR;MR1;IF RDC    %GET Y-ADDRESS;SAR=W VALUE                  0460CC00 D
018D  01F0 0000 003C C0E0          31=LIT                                                                  0461CC00 D
018E  AC08 0000 0C3C 40F0          WHEN RDC THEN CSAR,BEX      %SAR= NOT W;B=Y ADDRESS                     0462CC00 D
018F  4809 0018 9C00 00F0          B111 R=A3                   %A3=MASK W BITS LONG                        0463CC00 D
0190  A809 0C43 0D1C CCF0          B=MAR2,BMI;MR2;IF RDC       %READ (Y);B=DATA WORD                       0464CC00 D
0191  4809 EC56 003C C0F0          A3 AND B=MIR                %MIR = MASKED DATA LSB                      0465CC0C D
0192  AC08 A0C0 0C00 80FC          WHEN RDC THEN A1=SAR,BEX    %B=(Y);SAR = P VALUE                        0466CC00 D
0193  4809 0C41 8B3C 40FC          B C=B,CSAR                  %B=(Y) SHIFTED BY P                         0467CC0C D
0194  4809 EC44 0F3C 00F0          A3 NRI B=BBI                %B=(Y) MASKED OR DATA                       0468CC0C D
0195  4809 0C41 8080 00F0          B C=MIR                     %MIR=(Y) OR DATA BACK TO POSITION           0469CC00 D
0196  9809 0CC0 003C 18F0          MW1;IF SAI                                                              0470CC0C D
0197  9828 0D00 0000 00FC          WHEN SAI THEN JUMP          %WRITE NEW (Y)                              0471CC00 D
                                STORESEO:  %SEO CHAR ADDR.;B=ICW,A3=OC/W/P                                0472CC00 D
```

```
0198  4809 0C40 9000 00F0          B R=A3                    %B=ICW;A3=0C/W/P                  0473CC00 D
0199  81F0 00C0 0C00 00A0          20=SAR;31=LIT                                               04740C00 D
019A  4809 E156 4000 00F0          A3 AND LIT=A1             %A1=P-VALUE                       0475CC00 D
019B  4809 E0C0 9000 00F0          A3 R=A3                   %A3=0C/W                          04760C00 D
019C  9000 0000 0000 0000          5=SAR                                                       04770C0C D
019D  4809 E156 0B00 80F0          A3 AND LIT=B,SAR          %B=SAR=W                          04780C00 D
019E  4809 AC5E 4038 40F0          A1-B=A1,CSAR,LMAR         %P-W=P=A1;SAR=COMP W;MAR=Y LOCATOR 0479CC00 D
019F  01B0 0000 0000 00E0          YADDR=LIT                                                   0480CC00 D
01A0  7409 A0DE 4000 00F0          IF NOT AOV THEN A1-1=A1                                     04810C00 D
01A1  4FC9 2C5E 4000 00F0          IF MST THEN LIT-B=A1;SET LC1  %A1=32-W=P                    04820C00 D
01A2  7409 A0DE 4000 00F0          IF NOT AOV THEN A1-1=A1                                     04830C00 D
01A3  A809 0C18 900C C8FC          B111 R=A3;MR1;IF RDC      %A3=MASK W BITS LONG              04840C00 D
01A4  AC08 0000 0C00 C0F0          WHEN RDC THEN BEX         %B=Y-ADDRESS                      04850C00 D
01A5  2FD9 0C46 0D1C 00F0          IF LC1 THEN B+1=MAR2,BMI;SET LC1;SKIP                       04860C00 D
01A6  4809 0C40 0D1C C0F0          B=MAR2,BMI                %B=DATA;MAR2=Y OR Y+1 ADDRESS     04870C00 D
01A7  A809 A000 0000 8CF0          A1=SAR;MR2;IF RDC         %SAR=NEW P; READ (Y)              04880C00 D
01A8  AC08 EC56 0C30 00F0          WHEN RDC THEN A3 AND B=MIR,BEX  %B=(Y) SHIFTED BY P         0489CC00 D
01A9  4809 EC44 0F00 C0F0          A3 NRI B=BBI              %B=(Y) MSAKED ORED WITH DATA      049CC00 D
01AA  4809 0C41 8030 C0F0          B C=MIR                   %RESTORE TO POSITION AND OUTPUT   04910C00 D
01AB  9809 0000 0C00 1CF0          MW2;IF SAI                                                  0492CC00 D
01AC  9C08 A0C1 0058 00F0          WHEN SAI THEN A1 L=MIR,LMAR  %P LEFT JUSTIFIED TO POSITION  0493CC00 D
01AD  81AC 0000 0000 0090          IAW=LIT;COMP 20=SAR                                         0494CC00 D
01AE  A809 2001 1000 C8F0          LIT L=A3;MR1;IF RDC       %READ ADDRESS OF ICW              0495CC00 D
01AF  01F0 00C0 0030 C0E0          31=LIT                                                      04960C00 D
01B0  AC08 00C0 0C38 C0F0          WHEN RDC THEN BEX,LMAR    %READ HOME ADDRESS OF ICW         0497CC00 D
01B1  01CC 00C0 0030 00E0          IAWADDR=LIT               %ADDRESS OF ADDRESS OF IAW        04980C00 D
01B2  A809 EC56 0F0C 08F0          A3 AND B=BBI;MR1;IF RDC   %B=NEW ICW                        04990C00 D
01B3  AC08 0C40 0C30 00F0          WHEN RDC THEN B=MIR,BEX   %MIR=NEW ICW;B=ADDRESS OF OLD ICW 0500C0C00 D
01B4  4809 0C40 001C 00F0          B=MAR2                    %ADDRESS OF OLD ICW               05010C00 D
01B5  9809 0000 0C00 1CF0          MW2;IF SAI                                                  05020C00 D
01B6  9828 0000 0030 00F0          WHEN SAI THEN JUMP        %RETURN FROM WHERE CALLED         0503CC00 D
01B7  480C 0000 0030 00F0  KSTABLE:  WAIT                                                      0504CC00 D
01B8  0BAC 0000 0030 00C0          KS1-1=AMPCR. KS2-1=AMPCR. KS3-1=AMPCR                       05050C00 D
01B9  0BB0 0000 0030 00C0
01BA  0BCC 00C0 0C00 00CC
01BB  0BD0 0000 0000 00C0          KS4-1=AMPCR. KS5-1=AMPCR. KS6-1=AMPCR. KS7-1=AMPCR          0506CC00 D
01BC  0BEC 00C0 0C00 C0C0
01BD  0BF0 00C0 0C00 00C0
01BE  0C00 0000 0000 00C0
                           KS1:   %(A 15-0)=>Y 15-0                                            C5070C0C D
01BF  AC08 0000 0C00 C0F0          WHEN RDC THEN BEX         %Y31-16 UNCHANGED                 05080C00 D
01C0  4809 0C40 8B30 00F0          B R=B                                                       05090C00 D
01C1  000C 0000 0000 0020          16 = SAR                                                    0510CC00 D
01C2  4809 0C41 1D30 C0F0          B L=A3,BMI                %A3=Y31-16                        0511CC00 D
01C3  4809 0C41 0B30 00F0          B L=B                                                       0512CC00 D
01C4  4809 0C40 8B30 00F0          B R=B                                                       0513CC00 D
01C5  4809 EC5C 0090 00F0          A3 OR B=MIR               %(A 15-0)=>Y 15-0                 05140C00 D
01C6  0C10 0000 0000 0040          OUTPUT2NORET-1=MPCR                                         0515CC00 D
                           KS2:   %(A15-0)=>Y 31-16                                           0516CC00 D
01C7  AC08 0000 0C00 00F0          WHEN RDC THEN BEX         %Y 15-0 UNCHANGED                 0517C0C0 D
01C8  4809 0C41 0B30 00F0          B L=B                                                       0518CCC0 D
01C9  0000 0003 0C00 0020          16 = SAR                                                    05190C00 D
01CA  4809 0C40 9000 00F0          B R=A3,BMI                                                  05200C00 D
01CB  4809 0C41 0B30 00FC          B L=B                     %A3=Y 15-0                        05210C00 D
01CC  4809 EC5C 1090 C0FC          A3 OR B=A3,MIR                                              05220C00 D
01CD  0C20 00C0 0030 0040          OUTPUT2NORET-1=MPCR                                         0523CC00 D
                           KS3:   %(A31-0)=>Y(31-0)                                           0524CC00 D
01CE  AC08 0000 0C30 00F0          WHEN RDC THEN BEX                                           0525CC00 D
01CF  0C30 00C0 0030 C040          OUTPUT2NORET-1=MPCR                                         0526CC00 D
                           KS4:   %(A 7-0)=>Y 7-0                                             05270C00 D
```

```
01D0  AC08 0000 0C30 C0F0          WHEN RDC THEN BEX                                    05280C00 D
01D1  0000 0000 0030 0010          8=SAR                                                05290C00 D
01D2  4809 0C40 8B00 40F0          B R=B,CSAR                                           053C0C00 D
01D3  4809 0C41 1D30 40F0          B L=A3,BMI,CSAR                                      05310C00 D
01D4  4809 0C41 0B00 40F0          B L=B,CSAR                                           05320C00 D
01D5  4809 0C40 8B00 00F0          B R=B                                                05330C00 D
01D6  4809 EC5C 1090 00F0          A3 OR B=A3,MIR                                       05340C00 D
01D7  0C4C 0000 0C30 0040          OUTPUT2NORET-1=MPCR                                  05350C00 D
                          KS5:     %(A7-0)=>Y15-8                                       05360C00 D
01D8  AC08 20C1 1C30 40F0          WHEN RDC THEN LIT L = A3,CSAR,BEX                    05370C00 D
01D9  0FF0 0000 0000 00B0          255 = LIT; COMP 8 = SAR                              05380C00 D
01DA  4809 EC44 1D30 00F0          A3 NRI B=A3,BMI                                      05390C00 D
01DB  4809 0C41 0B30 00F0          B L=B                                                05400C00 D
01DC  4809 0C40 8B30 00F0          B R=B                                                05410C00 D
01DD  0000 0CC0 0030 C020          16=SAR                    %(A7-0) IN B(15-8)         05420C00 D
01DE  4809 EC5C 1090 00F0          A3 OR B=A3,MIR                                       05430C00 D
01DF  0C5C 00C0 0C30 0040          OUTPUT2NORET-1=MPCR                                  05440C00 D
                          KS6:     %(A7-0)=>Y23-16                                      05450C00 D
01E0  AC08 2001 1C30 00F0          WHEN RDC THEN LIT L=A3,BEX                           05460C00 D
01E1  0FFC 0000 0000 00A0          255=LIT;16=SAR                                       05470C00 D
01E2  4809 EC44 1D00 C0F0          A3 NRI B=A3,BMI                                      05480C00 D
01E3  4809 0C41 0B00 00F0          B L = B                                              05490C00 D
01E4  000C 00C0 0000 0010          COMP 24 = SAR                                        055C0C00 D
01E5  4809 0C40 8B30 00F0          B R=B                                                0551CC00 D
01E6  4809 EC5C 1C90 00F0          A3 OR B=A3,MIR                                       0552CC00 D
01E7  0C6C 0000 0C00 0040          OUTPUT2NORET-1=MPCR                                  0553CC00 D
                          KS7:     %(A7-0)=>Y31-24                                      05540C00 D
01E8  AC08 0000 0C0C 00F0          WHEN RDC THEN BEX                                    0555CC00 D
01E9  4809 0C41 0B30 40F0          B L=B,CSAR                                           0556C0C0 D
01EA  0000 0000 0C30 C030          COMP 8 = SAR                                         0557CC00 D
01EB  4809 0C40 9D00 00F0          B R=A3,BMI                %A3=Y23-0                  0558C0C0 D
01EC  4809 0C41 0B30 00FC          B L = B                                              0559CC00 D
01ED  48C9 EC5C 1C90 00F0          A3 OR B=A3,MIR                                       0560CC00 D
01EE  0C7C 0000 0030 0040          OUTPUT2NORET-1=MPCR                                  0561CC00 D
                          %                                                             0562CC00 D
                          %                                                             0563CC00 D
                          % ######################    SUBR. FOR I/O FUNCTIONS    ########0564CC30 D
                          IODUMPREG:                                                    0565CC00 D
01EF  4809 0000 0C90 00F0          0 = MIR                                              0566CC0C D
01F0  0C8C 0000 0000 0040          SIGLOADER-1=MPCR                                     0567CC00 D
                          IODUMPADR:  %B=ADDRESS TO PRINT                               0568CC00 D
01F1  4809 03C1 1030 00F0          1 L = A3                                             0569CC00 D
01F2  0000 00C0 0030 C020          COMP 16 = SAR                                        0570CC00 D
01F3  4809 EF5C 0C90 00F0          A3 OR BMAR=MIR            %OUTPUT LAST Y WRITTEN     0571CC00 D
01F4  0C90 0000 0000 0040          SIGLOADER-1=MPCR                                     0572CC00 D
                          CARDREAD:  %READ A CARD INTO ADDRESS IN B                     0573CC00 D
01F5  4809 2001 1C00 00F0          LIT L=A3                  %FUNCTION CODE IS 2/ADDRESS 0574CC00 D
01F6  0020 0000 0000 C0A0          2=LIT;COMP 16=SAR                                    0575CC0C D
01F7  4809 EC5C 0090 00F0          A3 OR B=MIR                                          0576CC00 D
01F8  0CA0 0000 0030 0040          SIGLOADER1-1=MPCR         %NO RETURN                 0577CC00 D
                          PRINTLINE:  %PRINT AN OUTPUT BUFFER ADDRESSED IN B            0578CC00 D
01F9  4809 2001 1030 00F0          LIT L=A3                  %FUNCTION CODE IS 7/ADDRESS 0579CC00 D
01FA  0070 0000 0000 C0A0          7=LIT; COMP 16=SAR                                   0580CC30 D
01FB  4809 EC5C 0C90 00F0          A3 OR B=MIR                                          0581CC00 D
01FC  0C8C 00C0 0030 0040          SIGLOADER1-1=MPCR                                    0582CC00 D
                          DISKREAD:   %READ/WRITE DISK USING INFO AT Y+(B)+(S)          0583CC00 D
                          %B=NUMBER OF WORDS                                            0584CC00 D
                          %B+1=DISK SECTOR TO BE ACCESSED                               0585CC00 D
                          %B+2= 3=>READ; 4=>WRITE                                       0586CC00 D
                          %B+3=ADDRESS OF BUFFER                                        0587CC00 D
```

```
01FD  4809 0C43 001C C0FC          B=MAR2                                                                  0588CC00 D
01FE  A809 0000 00D0 0CF0          MR2;IF RDC                                                              0589CC00 D
01FF  AC08 0F46 0C1C C0F0          WHEN RDC THEN BMAR+1=MAR2,BEX                                           0590CC00 D
0200  A809 0C41 1030 CCFC          B L=A3;MR2;IF RDC            %A3=NUM WORDS/0                            0591CC00 D
0201  0000 0000 0000 0020          COMP 16=SAR                                                             0592CC00 D
0202  AC08 0F46 0C1C C0F0          WHEN RDC THEN BMAR+1=MAR2,BEX                                           0593CC00 D
0203  A809 EC5C 1000 0CF0          A3 OR B=A3;MR2;IF RDC        %A3=NUM WORDS/SECTOR                       0594CC00 D
0204  AC08 0F46 0C1C 00F0          WHEN RDC THEN BMAR+1=MAR2,BEX                                           0595CC00 D
0205  A809 0C41 4030 0CF0          B L=A1;MR2;IF RDC            %A1=3 OR 4/0                               0596CC00 D
0206  AC08 00D8 001C C0F0          WHEN RDC THEN B11C=MAR2                                                 0597CC00 D
0207  4809 E000 0090 00F0          A3=MIR                                                                  0598CC00 D
0208  9809 AC5C 4030 1CFC          A1 OR B=A1;MW2;IF SAI        %A1=3-4/ADDRESS                            0599CC00 D
0209  9CD8 ACC0 0C90 00F0          WHEN SAI THEN A1=MIR                                                    0600CC00 D
020A  0A5C 0CC0 0C30 00C0    SIGLOADER1:  IFETCH-1=AMPCR                                                  0601CC00 D
                             SIGLOADER:                                                                   0602CC00 D
020B  4809 0019 0C1D 00F0          B111=MAR2,CTR                                                           0603CC00 D
020C  9809 0000 C000 1CFC          MW2; IF SAI                                                             0604CC00 D
020D  1AC6 00C0 0CD0 00F0          IF GC2 THEN RESET GC2; STEP ELSE SKIP                                   0605CC00 D
020E  49C9 0000 0000 00FC          SET LC1                      %SIGNALS GC2 MUST BE RESET                 0606CC00 D
020F  9B48 00C0 0C30 00F0          WHEN SAI THEN RESET GC1                                                 0607CC00 D
0210  8AC8 0000 0092 00F0          INC;WHEN COV THEN RESET GC2;STEP                                        0608CC00 D
0211  0988 0000 0000 C0F0          SET GC1;WHEN GC1 THEN STEP                                              0609CC00 D
0212  280D 0000 0090 00F0          IF LC1 THEN STEP ELSE JUMP                                              0610CC00 D
0213  18A8 0000 0000 00F0          SET GC2; WHEN GC2 THEN JUMP                                             0611CC00 D
                             %                                                                            0612CC00 D
                             ERRORMSG:  %B=NUMBER OF ERROR MESSAGE TO BE PRINTED                          0613CC00 D
                                    %0=ILLEGAL INSTRUCTION                                                0614CC00 D
                                    %1=INSTRUCTION NOT IMPLEMENTED                                        0615CC00 D
                                    %2=SQUARE ROOT OF NEGATIVE NUMBER                                     0616CC00 D
0214  4809 0C41 0BD0 00F0          B L=B                        %PRINT ERROR MESSAGE                       0617CC0C D
0215  005C 0000 0030 C0AC          16=SAR;5=LIT                                                            0618CC00 D
0216  4809 2C5D 8080 00F0          LIT OR B C=MIR               %MIR= 5/B                                  0619CC00 D
0217  20AC 0000 0090 C060          SIGLOADER-1=CFCR                                                        0620CC00 D
0218  4809 C0C1 0B30 00F0          A2 L=B                                                                  0621CC00 D
0219  0060 00D0 0030 00E0          6=LIT                        %OUTPUT ADDRESS AND CONTENTS               0622CC0C D
021A  3819 2C5D 8090 00FC          LIT OR B C=MIR;IF LC2 THEN SKIP                                         0623CC00 D
021B  2090 00C0 C030 0040          SIGLOADER1-1=MPCR            %RETURNS TO IFETCH                         0624CC00 D
021C  0BC0 0000 0090 C0C0          HALFFETCH=AMPCR                                                         0625CC0C D
021D  20A0 0000 0090 0040          SIGLOADER-1=MPCR             %RETURNS TO HALFFETCH                      0626CC00 D
021E  481B 0000 0B00 00F0    ILLEGAL:  0=B;SKIP                 %RETURNS VIA IFFETCH                       0627CC00 D
021F  4849 0000 0B30 00FC    ILLEGAL4:   0=B;SET LC2            %PRINT ILLEGAL INSTRUCTION                 0628CC00 D
0220  2130 0000 0090 0040          ERRORMSG-1=MPCR              %ILLEGAL TYPE IV;RETURN TO HALFFETCH       0629CC00 D
0221  485B 2000 0BD0 C0F0    ILLEGALCMR1:  LIT=B;SET LC2;SKIP                                             0630CC00 D
0222  4809 20C0 0B30 00F0    ILLEGALCMR:  LIT=B                                                           0631CC00 D
0223  0030 0C00 0000 C0E0          3=LIT                        %ERROR 3=>ILLEGAL CMR REFERENCE            0632CC00 D
0224  213C 00C0 0090 0040          ERRORMSG-1=MPCR                                                         0633CC00 D
0225  481B 00C0 0B00 00F0    UNWRITTEN:  1=B;SKIP                                                         0634CC00 D
0226  4849 00C0 0B00 00F0    UNWRITTEN4:  1=B;SET LC2           %PRINT INSTRUCTION NOT IMPLEMENTED         0635CC00 D
0227  2130 0000 0090 0040          ERRORMSG-1=MPCR              %PRINT AND RETURN TO HALFFETCH             0636CC0C D
                             %                                                                            0637CC00 D
                             % ********************     SUBR. FOR REGISTER STORES     **************      0638CC00 D
                             PUTPAR:                           %SAVE CONTENTS OF PAR REGISTER IN 8        0639CC00 D
0228  4809 C0C0 0038 C0F0          A2=MIR,LMAR                                                             0640CC00 D
0229  01DC 0000 0000 C0E0          PAR=LIT                                                                 0641CC00 D
022A  0CCC 00C0 0090 0040          OUTPUT1-1=MPCR                                                          0642CC00 D
                             GETPAR:                           %RESTORE CONTENTS OF PAR FROM 8            0643CC00 D
022B  4809 0000 0C38 00F0          LMAR                                                                    0644CC00 D
022C  01DC 00C0 0000 00E0          PAR=LIT                                                                 0645CC0C D
022D  A809 0000 0090 08FC          MR1;IF RDC                                                              0646CC00 D
022E  ACC8 0000 0CD0 00F0          WHEN RDC THEN BEX                                                       0647CC0C D
```

```
022F  482D 0040 2030 00F0          B=A2;JUMP                                                                  06480C00 D
                                OUTPUT:   %NO RETURN                                                         06490C00 D
0230  0A50 0000 0000 00C0          IFETCH-1=AMPCR                                                             0650000C D
0231  9809 0000 0030 18F0       OUTPUT1:  MW1;IF SAI                                                          06510C00 D
0232  9C28 0000 0000 00F0          WHEN SAI THEN 0;JUMP                                                       0652C000 D
0233  0A5C 0000 0C30 C0C0    OUTPUT2NORET:  IFETCH-1=AMPCR                                                    06530C00 D
0234  9809 000D 0030 1CF0       OUTPUT2:  MW2;IF SAI                                                          06540C00 D
0235  9C08 0000 0030 00F0          WHEN SAI THEN 0;STEP                                                       06550C00 D
0236  B80D 0000 0030 00F0          IF IRQ THEN STEP ELSE JUMP                                                 06560C00 D
0237  1F0C 0000 0030 0040          IODUMPADR-1=MPCR                                                           0657CC00 D
                                YOUT:     %Y-ADDRESS IN MAR2 CAME FROM PRECALCULATED EXCEPT REP              06580C00 D
0238  1C19 C000 9030 00F0          IF GC2 THEN A2 R=A3;SKIP   %REPLACE REPEAT IS SPECIAL                      06590000 D
0239  2320 00C0 0030 0040          OUTPUT2NORET-1=MPCR                                                        06600C00 D
023A  81B0 0000 0C30 C0A0          20=SAR;YADDR=LIT                                                           06610C00 D
023B  4809 E000 0030 00F0          A3                                                                         06620C00 D
023C  5C19 0000 0008 00F0          IF LST THEN LMAR;SKIP      %CHECK FOR REPEAT INSTR. MODE                   0663CC00 D
023D  2320 00C0 0030 0040          OUTPUT2NORET-1=MPCR                                                        06640C00 D
023E  A809 0000 0030 C8FC          MR1;IF RDC                 %READ Y-ADDRESS                                 06650C00 D
023F  AC08 00C0 0C3C 00F0          WHEN RDC THEN BEX                                                          0666CC00 D
0240  3C09 0C40 0C1C 00F0          IF LC2 THEN B=MAR2                                                         0667CC00 D
0241  2320 00C0 0030 0040          OUTPUT2NORET-1=MPCR                                                        06680C00 D
                                % ########################    LOGICAL SUBROUTINES          ##############06690C00 D
                                OUTLOGIC:                     %Y ADDRESS IN B,A ADDR IN MAR1                 06700C00 D
0242  2A59 000D 0030 00F0          IF LC1 THEN SET LC2;SKIP                                                   06710C00 D
0243  22F0 0000 0030 C040          OUTPUT-1=MPCR              %OPCODE01=>  NO RETURN                          06720C00 D
0244  230C 0000 0000 006C          OUTPUT1-1=CPCR             %OPCODE 02 => RETURN                            06730C00 D
0245  4809 A00D 001C 00F0          A1=MAR2                    %SAVE Y-ADDRESS IN A1                           06740C00 D
0246  2370 0000 0000 004C          YOUT-1=MPCR                %OUTPUT TO Y ADDRESS                            0675CC00 D
0247  4809 A000 9C00 0CF0    GETF:     A1 R=A3                %ISOLATE AND RETURN F FIELD                     06760000 D
0248  B070 00C0 0030 00A0          20=SAR;7=LIT                                                               06770C00 D
0249  482D E156 1000 00FC          A3 AND LIT=A3;JUMP                                                         06780C00 D
024A  4809 A0C0 8B30 00F0    GETAB:    A1 R=B                 %ISOLATE AND RETURN IN B & MAR A-FLD            06790C00 D
024B  B070 0000 0C30 00A0          23=SAR;7=LIT                                                               06800C00 D
024C  482D 2C56 0B3C 00F0          LIT AND B=B,MAR;JUMP                                                       06810C00 D
024D  4809 A000 9030 00F0    GETAA3:   A1 R=A3                %ISOLATE AND RETURN IN A3&MAR A-FIELD           06820C00 D
024E  B070 000D 0000 00A0          23=SAR;7=LIT                                                               06830000 D
024F  482D E156 103C 00F0          A3 AND LIT=A3,MAR;JUMP                                                     0684CC00 D
                                GETF1:                        %RETURNS F1 FIELD FOR TYPE 3 IN A3             0685CC00 D
0250  4809 A00D 9030 00F0          A1 R=A3                                                                    06860000 D
0251  9030 0000 0000 00AC          21=SAR;3=LIT                                                               0687CC00 D
0252  482D E156 1000 00F0          A3 AND LIT=A3;JUMP                                                         06880C00 D
                                CHECKCMR:  %CMR ADDRESS IN A3;LC1 SET IF PRIVILEGED;RETURN LC1 IF LEGAL      06890C00 D
0253  4809 E15E 0C00 00F0          A3 LSS LIT                 %CHECK ADDR LT 20 OCTAL                         06900C00 D
0254  010C 0000 0030 00EC          16=LIT                                                                     06910C00 D
0255  73E9 0000 0030 00F0          IF TRUE THEN SET LC1; JUMP                                                 0692CC00 D
0256  4809 E15E 003D C0F0          A3 LSS LIT                 %CHECK ADDRESS LT 30 OCTAL                      0693C000 D
0257  0180 00C0 0000 C0E0          24=LIT                     %CHECH A:DR LT 30 OCTAL                         0694C000 D
0258  7C19 E15E 0C00 00FC          IF FALSE THEN A3 LSS LIT; SKIP                                             06950C00 D
0259  2BED 0000 0030 00FC          JUMP; IF LC1 THEN SET LC1                                                  06960C00 D
025A  0300 0000 0000 00E0          48=LIT                     %CHECK ADDR LT 60 OCTAL                         06970C00 D
025B  7C19 E15E 0C00 00F0          IF FALSE THEN A3 LSS LIT;SKIP                                              06980CC0 D
025C  282D 00C0 0C30 00FC          JUMP; IF LC1                                                               06990C00 D
025D  0580 00C0 0000 00E0          88=LIT                                                                     0700CC00 D
025E  7C19 E15E 0000 C0F0          IF FALSE THEN A3 LSS LIT;SKIP                                              07010C00 D
025F  2BED 00C0 0030 00F0          JUMP; IF LC1 THEN SET LC1                                                  07020C00 D
0260  060C 00C0 0030 00EC          96=LIT                                                                     0703CC00 D
0261  7C19 E15E 003C C0F0          IF FALSE THEN A3 LSS LIT;SKIP                                              0704CC00 D
0262  282D 0000 0030 00F0          JUMP; IF LC1                                                               0705CC00 D
0263  280C 0000 0030 C090          128=LIT;10=SAR                                                             07060C00 D
0264  7C19 000D 0030 00F0          IF TRUE THEN SKIP                                                          0707CC00 D
```

```
0265  282D 00C0 0C30 00F0          JUMP; IF LC1                                                    0708CC00 D
0266  2BED 0000 0C30 00FC          JUMP;IF LC1 THEN SET LC1                                        0709CC00 D
                                % ************************    JUMP SUBROUTINES            ************0710CC00 D
                                PUTJUMP:    %A3=JUMP ADDRESS;STORE IN A3&PAR                      0711CC00 D
0267  2C19 C000 A030 C0F0          IF LC1 THEN A2 R=A2;SKIP                                        0712CC00 D
0268  0A50 0000 0030 0040          IFETCH-1=MPCR              %NO JUMP IF NOT LC1                  0713CC00 D
0269  000C 0000 0000 0020          16=SAR                                                          0714CC00 D
026A  3809 C001 0B30 00F0          A2 L=B;IF LC2                                                   0715CC0C D
026B  4809 EC5C 2030 00F0          A3 OR B=A2                 %UPDATE PAR                          0716CC00 D
026C  2270 0000 0C30 0060          PUTPAR-1=CPCR              %WRITE TO PAR LOCATION AND RETURN    0717CC00 D
026D  0A50 00C0 0030 0040          IFETCH-1=MPCR                                                   0718CC00 D
                                %                                                                  0719CC00 D
                                %                                                                  0720CC00 D
                                % ************************    SHIFT SUBROUTINES           ************0721CC00 D
                                SHIFTAMOUNT:   %RETURNS B=(A-REG);MAR=A-ADDR.;SAR=SHIFTAMOUNT      0722CC00 D
026E  4809 A0C1 1030 00F0          A1 L=A3                                                         0723CC00 D
026F  2000 0CC0 0030 0030          COMP 6=SAR                 %DROP OPCODE FIELD                   0724CC00 D
0270  4809 E000 9030 00F0          A3 R=A3                    %ISOLATE A/M FIELDS IN LOW 10 BITS   0725CC00 D
0271  A000 00C0 0030 0020          22=SAR                                                          0726CC00 D
0272  4809 E156 803C 00FC          A3 AND LIT R=MAR           %MASK B-FIELD INTO MAR               0727CC00 D
0273  10E0 0000 0030 0080          14=LIT;1=SAR                                                    0728CC00 D
0274  4809 E156 8B00 00F0          A3 AND LIT R=B             %B=BITS 5&6                          0729CC00 D
0275  960C 0000 0030 0080          96=LIT;5=SAR               %MASK FOR BITS 5&6                   0730CC00 D
0276  4809 2C52 0C00 20FC          LIT EQL B;ASR                                                   0731CC00 D
0277  002C 0000 0000 00E0          2=LIT                      %2=> SHIFT AMOUNT IN B(B)            0732CC00 D
0278  6FC9 2F40 003C 00F0          IF TRUE THEN LIT+6MAR=MAR;SET LC1                               0733CC00 D
0279  008C 00C0 0000 C0E0          BBASE=LIT                                                       0734CC0C D
027A  A809 2C52 0030 00F0          LIT EQL B;IF RDC                                                0735CC00 D
027B  003C 0000 003C 00E0          3=LIT                      %3=> SHIFT AMOUNT IN A(B)            0736CC00 D
027C  6BC9 00C0 0030 00F0          IF TRUE THEN SET LC1       %MAR SETUP FOR A(B)                  0737CC00 D
027D  2A0B 0000 0000 C8F0          IF LC1 THEN MR1;STEP ELSE SKIP                                  0738CC00 D
027E  AC18 0000 0C30 00F0          WHEN RDC THEN BEX;SKIP     %B=A(B) OR B(B)                      0739CC00 D
027F  4809 E156 0B00 00FC          A3 AND LIT=B               %B=M-FIELD                           0740CC00 D
0280  B3F0 0000 0C30 0080          63=LIT;7=SAR               %MASK M-FIELD OF 63                  0741CC00 D
0281  4809 E000 803D 00F0          A3 R=MAR,CTR               %SAVE A-ADDRESS IN CTR               0742CC00 D
0282  A809 2C56 1000 88F0          LIT AND B=SAR,A3;MR1;IF RDC                                     0743CC00 D
0283  4809 E15E C000 00F0          A3 GEQ LIT                                                      0744CC00 D
0284  020C 0000 003C C0E0          32=LIT                                                          0745CC00 D
0285  7FC9 E15E 0C00 80F0          IF TRUE THEN A3-LIT=SAR;SET LC1                                 0746CC00 D
0286  AC28 00C0 0C00 00F0          WHEN RDC THEN BEX;JUMP     %D=(A-REG);MAR=A-ADDR;SAR=SHIFT AMT. 0747CC00 D
                                %                                                                  0748CC00 D
                                %                                                                  0749CC00 D
                                % ************************    COMPARE SUBROUTINES         ************0750CC00 D
                                SETCOMPARE:    %COMPARE A(B) WITH A(A)  AND SET CD BITS            0751CC00 D
0287  0BCC 00C0 C030 C0C0          HALFFETCH=AMPCR                                                 0752CC00 D
0288  280B 00C0 0000 00FC          IF LC1 THEN STEP ELSE SKIP %LC1=> CALLED BY OPR 44,46,47        0753CC00 D
0289  0A50 0000 0030 00C0          IFETCH-1=AMPCR             %ELSE OPR 74X4,6,7                   0754CC00 D
                                SETCDRET:  %ENTER HERE AND RETURN WHERE CALLED FROM                0755CCC0 D
028A  4849 EC52 0C00 00F0          A3 EQL B;SET LC2           %EQUALITY CK;SETUP FOR REPEAT        0756CC00 D
028B  6C19 D81D A030 40FC          IF TRUE THEN A2 OR B100 C=A2,CSAR;SKIP  %SET EQUAL BIT          0757CC00 D
028C  4809 D80F AC30 40F0          A2 AND B011 C=A2,CSAR      %CLEAR EQUAL BIT                     0758CC0C D
                                   %GREATER THEN EQUAL TO BIT NOW IN MSB                           0759CC00 D
028D  B000 0000 0C30 0030          31=SAR                                                          0760CC00 D
028E  4809 EC4C 0C30 00F0          A3 XOR D                                                        0761CC0C D
028F  480B EC5E 0000 C0F0          A3 LSS B; IF MST THEN STEP ELSE SKIP                            0762CC00 D
0290  7C2B D80F AC30 00F0          IF FALSE THEN A2 AND B011 C = A2; JUMP ELSE SKIP                0763CC00 D
0291  7429 D8CF A030 00F0          IF TRUE THEN A2 AND B011 C=A2;JUMP                              0764CC00 D
0292  482D D81D A00C C0FC          A2 OR B100 C=A2;JUMP                                            0765CC00 D
                                   %A2 RESTORED TO NORMAL BEFORE EXIT TO WHERE CALLED FROM          0766CCCC D
                                %                                                                  0767CC00 D
```

```
                                                %                                                                    0768CC00 D
                                                % ************************    BIT COUNT SUBROUTINE        ************0769CC0C D
                                                COUNTONES:   %COUNT NUMBER OF ONES IN B AND RETURN COUNT IN A3,MIR    07700C00 D
0293  4809 0300 1C01 00F0                               0=A3,LCTR                                                      07710C00 D
0294  11FC 0000 0000 0080                               31=LIT;1=SAR                                                   07720C00 D
0295  4809 0C41 8B02 00F0                      CHECKLSB:  B C=B,INC                                                    07730C00 D
0296  5C09 E0C0 1030 00F0                               IF LST THEN A3+1=A3,MIR                                        07740C00 D
0297  8829 00C0 0030 00F0                               IF COV THEN JUMP                                               07750C00 D
0298  294C 0000 0030 0040                               CHECKLSB-1=MPCR                                                07760C00 D
                                                %                                                                    07770C00 D
                                                %                                                                    07780C00 D
                                                % ************************    MULTIPLY SUBROUTINE         ************07790C00 D
                                                MULTIPLY:  %A2 AND B CONTAIN OPERATORS                                07800C0C D
0299  4809 0000 5001 00F0                               0=A1,A3,LCTR                                                   07810C00 D
029A  11F0 0000 0030 0080                               31=LIT;1=SAR                                                   07820C00 D
029B  2809 0C52 0000 00F0                               0 EOL B;IF LC1              %TEST FOR ZERO MULTIPLY            07830C00 D
029C  640D CC4C 0C00 00FC                               IF FALSE THEN A2 XOR B;STEP ELSE JUMP                          07840C00 D
029D  4BC9 C000 0C30 00F0                               A2;IF MST THEN SET LC1      %TEST FOR DIFFERENT SIGNS          07850C00 D
029E  4C09 C0C2 2030 00F0                               IF MST THEN NOT A2=A2       %TEST FOR NEGATIVE                 07860C00 D
029F  4809 0C40 0C30 00F0                               B                           %TEST FOR NEGATIVE                 07870C00 D
02A0  4C09 0C42 0B0C 00F0                               IF MST THEN NOT B=B                                            07880C00 D
02A1  4809 C0C0 A032 00F0                      MULT1:   A2 R=A2,INC                                                    07890C00 D
02A2  5C19 AC40 C000 00F0                               IF LST THEN A1+B R=A1;SKIP %A1=MOST SIGNIFICANT                07900C00 D
02A3  4809 A000 C030 00F0                               A1 R=A1                                                        07910C00 D
02A4  5C09 F81C 1C30 00F0                               IF LST THEN A3 OR B100=A3  %A3=LEAST SIGNIFICANT               07920C00 D
02A5  840B E000 9000 C0F0                               IF NOT COV THEN A3 R=A3;STEP ELSE SKIP                         07930C00 D
02A6  2A00 0000 003C 0040                               MULT1-1=MPCR                                                   07940C00 D
02A7  2C0D E002 1000 00F0                               IF LC1 THEN NOT A3=A3;STEP ELSE JUMP                           07950C00 D
02A8  482D A002 4030 00F0                               NOT A1=A1;JUMP                                                 07960C00 D
                                                %                                                                    07970C00 D
                                                %                                                                    07980C00 D
                                                % ************************    DIVIDE SUBROUTINE           ************07990C00 D
                                                DIVIDE:  %B=DIVISOR; A3=MOST DIVIDEND; MAR=LEAST DIVIDEND ADDRESS     08000C0C D
02A9  4849 0000 6001 00F0                               0=A1,A2,LCTR;SET LC2        %0=>A2; REM.=>A1                   08010C00 D
02AA  31E0 0000 0C30 00B0                               30=LIT;COMP 1=SAR                                              08020C00 D
02AB  4809 0C52 0000 C0F0                               B EOL 0                                                        08030C00 D
02AC  6BE9 E0C3 0C30 C0F0                               A3; IF TRUE THEN SET LC1;JUMP %OVERFLOW=>DIVISION BY ZERO       08040C00 D
02AD  4FC9 E002 1000 00F0                               IF MST THEN NOT A3=A3;SET LC1                                  08050C00 D
02AE  A809 0C40 0030 08F0                               B;MR1;IF ROC                %READ LEAST DIVIDEND               08060C0C D
02AF  4C09 1458 0B00 00F0                               IF MST THEN BIFF=B          %DIVISOR NEGATIVE                  08070C00 D
02B0  4809 E001 1030 00F0                      DIV1:    A3 L=A3                                                        08080C00 D
02B1  4C09 A0DC 403C 00F0                               IF MST THEN A1 OR B001=A1                                      08090C00 D
02B2  4809 A45E 0C00 00F0                               A1 LSS BOTT                                                    08100C00 D
02B3  7C0B C0DC 2030 C0FC                               IF FALSE THEN A2 OR B001=A2;STEP ELSE SKIP                     08110C00 D
02B4  4809 A45E 4030 C0F0                               A1-BOTT=A1                                                     08120C00 D
02B5  8019 00C0 0002 C0F0                               INC;IF NOT COV THEN SKIP                                       08130C00 D
02B6  0CDC 0000 0C00 0040                               DIV2-1=MPCR                                                    08140C00 D
02B7  4809 A001 4C30 00F0                               A1 L=A1                                                        08150C00 D
02B8  4809 C001 2030 00F0                               A2 L=A2                                                        08160C00 D
02B9  4BE9 0000 0030 C0F0                               IF MST THEN SET LC1;JUMP                                       08170C00 D
02BA  2AFC 0000 003C 0040                               DIV1-1=MPCR                                                    08180C00 D
02BB  3C19 C0C1 2001 00F0                      DIV2:    IF LC2 THEN A2 L=A2,LCTR;SKIP                                  08190C00 D
02BC  0CE0 00C0 0000 004C                               DIV3-1=MPCR                                                    08200C00 D
02BD  4809 0C40 1000 00F0                               B=A3                                                           08210C00 D
02BE  AC08 A001 4C3C C0F0                               WHEN ROC THEN A1 L=A1,BEX   %B=LEAST DIVIDEND                  08220C00 D
02BF  4809 EC4C 1000 C0F0                               A3 XOR B=A3                 %SWAP B AND A3                     08230C00 D
02C0  4809 EC4C 0B30 00F0                               A3 XOR B=B                                                     08240C00 D
02C1  4809 EC4C 1C30 C0F0                               A3 XOR B=A3                                                    08250C00 D
02C2  2FC9 E0C2 1030 00F0                               IF LC1 THEN NOT A3=A3;SET LC1                                  08260C00 D
02C3  2AFC 0000 0030 0040                               DIV1-1=MPCR                                                    08270C00 D
```

```
02C4  4809 CC40 0030 00F0      DIV3:  B                                %QUOTIENT IN A2;REM IN A1                0828CC00 D
02C5  4019 0000 0030 00F0             IF NOT MST THEN SKIP                                                      0829CC0C D
02C6  0CF0 0000 0C30 0040             DIVNEG-1=MPCR                                                             0830CC00 D
02C7  2C0D C0C2 2030 C0F0             IF LC1 THEN NOT A2=A2;STEP ELSE JUMP                                      0831CC00 D
02C8  482D A0C2 4C30 C0F0             NOT A1=A1;JUMP                                                            0832CC00 D
02C9  2C29 A002 4030 00F0      DIVNEG:  IF LC1 THEN NOT A1=A1;JUMP                                              0833CC00 D
02CA  482D C002 2030 C0F0             NOT A2=A2;JUMP                                                            0834CC00 D
                               %                                                                                0835CC00 D
                               %                                                                                0836CC00 D
                               % ******ROUTINES FOR REPEAT AND INDIRECTION CONTROL **************               0837CC30 D
                               GC2CHECK:                               % GC2 IS SET                              0838CC00 D
02CB  4809 C3C1 1C00 00F0             A2 L = A3                                                                 0839CC00 D
02CC  A000 0C00 0000 0020             COMP 10=SAR                       %CHECK BIT 21=REPEAT BIT                0840CC00 D
02CD  4809 E0C0 0030 C0F0             A3                                                                        0841CC0C D
02CE  480B 0000 0C0C 00FC             IF MST THEN STEP ELSE SKIP      % CHECK FOR REPEAT MODE                   0842CC00 D
02CF  0D00 0000 0C00 0060             RPTCHECK-1 = CPCR                                                         0843CC00 D
                               GC2CHECK1:                                                                       0844CC00 D
02D0  3809 2001 0B00 C0F0             LIT L = B; IF LC2                                                         0845CC00 D
02D1  3070 0003 0030 0090             7 = LIT; COMP 21 = SAR                                                    0846CC00 D
02D2  48C9 CC4E 2030 00F0             A2 NIM B = A2; RESET GC2        % CLEAR A2(23-21)                         0847CC00 D
02D3  0AA0 0000 0C30 0040             IFETCH1- 1 = MPCR                                                         0848CC00 D
                               RPTCHECK:                                                                        0849CC00 D
02D4  4809 00C0 0038 C0F0             LMAR                              % B(7)                                  0850CC00 D
02D5  00F0 0000 0030 00E0             15 = LIT                                                                  0851CC00 D
02D6  A809 00C0 0000 C8F0             MR1; IF RDC                                                               0852CC00 D
02D7  AC08 00C0 0C30 00FC             WHEN RDC THEN BEX                                                         0853CC00 D
02D8  4809 0C42 0C30 00F0             NOT B                                                                     0854CC00 D
02D9  640D 0C40 1000 00F0             IF NOT ABT THEN B=A3;STEP ELSE JUMP  %(B(7))=C,RETURN GCCHECK             0855CC00 D
02DA  4809 E0DE 0030 00F0             A3-1 = MIR                                                                0856CC00 D
02DB  230C 0000 0030 0060             OUTPUT1-1 = CPCR                  % DEC B(7)                              0857CC00 D
02DC  4809 0000 0038 00F0             LMAR                                                                      0858CC00 D
02DD  018C 00C0 0030 00E0             RINST=LIT                         %REPEAT INSTR.                          0859CC00 D
02DE  A809 0000 0030 08F0             MR1; IF RDC                                                               0860CC00 D
02DF  AC08 00C0 0C30 00FC             WHEN RDC THEN BEX                 % REPEAT INST.                          0861CC0C D
02E0  4809 0C40 C030 00F0             B R=A1,MIR                                                                0862CC00 D
02E1  B070 0003 0000 C0A0             7 = LIT; 23 = SAR                                                         0863CC00 D
02E2  4809 A156 4030 00F0             A1 AND LIT = A1                   % A-FIELD                               0864CC00 D
02E3  0D10 0000 0030 00C0             ACMP0-1 = AMPCR                   % CHECK A-FIELD                         0865CC00 D
02E4  3829 00C0 0030 00FC             IF LC2 THEN JUMP                  % FOR TERMINATE                         0866CC00 D
02E5  0D20 0C00 0030 0040             ANCMP0-1 = MPCR                   % CONDITIONS                            0867CC00 D
02E6  4809 00C0 0038 00F0      RPTCHECK1:  LMAR                                                                0868CC00 D
02E7  018C 00C0 0C30 00E0             RINST=LIT                                                                 0869CC0C D
02E8  A809 0000 C000 08F0             MR1;IF RDC                                                                0870CC00 D
02E9  AC08 00C0 0C30 00FC             WHEN RDC THEN BEX                                                         0871CC00 D
02EA  4809 0C41 0B38 00F0             B L = B,LMAR                                                              0872CC00 D
02EB  0190 0000 0C00 00A0             CINST=LIT;16=SAR                  %CURRENT INSTR.                         0873CC00 D
02EC  A809 0C40 0030 08FC             B;MR1;IF RDC                                                              0874CC0C D
02ED  4C19 0018 C000 00F0             IF MST THEN B111 R=A1;SKIP                                                0875CC00 D
02EE  4809 0000 4030 00F0             0=A1                                                                      0876CC00 D
02EF  AC08 AC50 8C8C 00F0             WHEN RDC THEN A1 OR B C=MIR;BEX                                           0877CC00 D
02F0  4809 0C40 4000 C0FC             B = A1                            % RESTORE C.I.                          0878CC00 D
02F1  4809 A000 9000 00F0             A1 R = A3                                                                 0879CC00 D
02F2  1070 00C0 0030 00A0             7 = LIT; 17 = SAR                                                         0880CC00 D
02F3  4809 E156 1D3C 00F0             A3 AND LIT = A3,BMI               % B-FIELD                               0881CC00 D
02F4  4809 E140 003C 00F0             A3 + LIT = MAR                                                            0882CC00 D
02F5  0080 0000 0030 00E0             BBASE = LIT                                                               0883CC00 D
02F6  A809 0000 0000 08F0             MR1; IF RDC                                                               0884CC00 D
02F7  AC08 0C40 1C00 00F0             WHEN RDC THEN B = A3,BEX                                                  0885CC00 D
02F8  4809 EC40 0030 00F0             A3 + B = MIR                                                              0886CC00 D
02F9  2300 00C0 003C C060             OUTPUT1-1 = CPCR                  % B(B)+SY=>B(B)                         0887CC00 D
```

```
02FA  4809 0000 0038 00F0          LMAR                                                    0888C000 D
02FB  01B0 0000 0030 00E0          YADDR=LIT                                               08890C00 D
02FC  A809 C000 0C00 08F0          MR1; IF RDC                                             08900C00 D
02FD  AC08 0000 0C30 00F0          WHEN RDC THEN BEX          % Y=Y+B(B)+S(6)              08910000 D
02FE  4809 EC40 0090 00F0          A3 + B = MIR               % Y + SY = Y                 08920C00 D
02FF  2300 0000 C030 0060          OUTPUT1-1 = CPCR                                        08930C00 D
0300  0B30 0000 0000 0040          IFETCH2 -1 = MPCR                                       08940C00 D
                                S6YFETCH:                     % Y,S,B => A1               08950C00 D
0301  4809 0640 0090 00FC          AMPCR = MIR                                             08960C00 D
0302  4809 A000 9030 00F0          A1 R = A3                                               08970C00 D
0303  1070 0000 0000 00A0          7 = LIT; 17 = SAR                                       08980C00 D
0304  4809 E156 1030 00F0          A3 AND LIT = A3                                         08990000 D
0305  4809 E140 003C 00F0          A3 + LIT = MAR             % B(B)                       C90CCC00 D
0306  0080 00C0 0030 00E0          BBASE = LIT                                             0901CC00 D
0307  A809 A0C1 0B00 48F0          A1 L = B,CSAR; MR1; IF RDC                              09020C00 D
0308  9160 0000 0030 0090          S6=LIT;COMP 19=SAR                                      09030C00 D
0309  AC08 0C40 9C3C 00F0          WHEN RDC THEN B R = A3,BEX   % B(B) => B               09040000 D
030A  4809 EC40 1008 C0F0          A3 + B = A3,LMAR             % Y +B(B)=> A3, S(6)=>MAR  09050C00 D
030B  A809 0C00 0030 08F0          MR1; IF RDC                                             09060C00 D
030C  01B0 0000 0030 00E0          YADDR=LIT                                               0907CC00 D
030D  AC08 0000 0C0C 00F0          WHEN RDC THEN BEX          % S(6)=>B                    09080C00 D
030E  4809 EC40 0D38 C0F0          A3+B=MIR,BMI,LMAR          %MIR=Y+B(B)+S(6)             09090000 D
030F  2300 0000 0C30 0060          OUTPUT1-1 = CPCR           % STORE Y AT 19              091C0C00 D
0310  4809 0C40 0090 00F0          B = AMPCR                                               09110C00 D
0311  4809 00C0 0000 C0F0          STEP                                                    0912CC00 D
0312  482D 0000 0030 00F0          JUMP                                                    09130C00 D
                                FETNORM:                                                  09140000 D
0313  4809 A000 9000 00F0          A1 R = A3                                               0915CC00 D
0314  1070 0000 0000 00A0          7 = LIT; 17 = SAR                                       09160C00 D
0315  4809 E156 1030 00F0          A3 AND LIT = A3                                         09170C00 D
0316  4809 E140 003C 00F0          A3 + LIT = MAR             % B(B)                       0918CC00 D
0317  0080 0003 0000 00E0          BBASE = LIT                                             09190C00 D
0318  A809 A0C1 0B00 48F0          A1 L = B,CSAR; MR1; IF RDC                              092C0C00 D
0319  91FC 0000 0000 0090          31 = LIT; COMP 19 = SAR                                 09210C00 D
031A  AC08 0C40 9C00 00F0          WHEN RDC THEN B R = A3,BEX    % Y=>A3,B(B)=>B           09220C00 D
031B  4809 EC40 1000 00F0          A3 + B = A3                % Y + B(B)=>A3               09230C00 D
031C  4809 A000 8B00 00F0          A1 R = B                                                09240C00 D
031D  9070 00C0 0030 0090          7 = LIT; 13 = SAR                                       09250C00 D
031E  4809 2C56 0B30 00F0          LIT AND B = B              % S-FIELD                    0926CC00 D
031F  4809 2C40 003C C0F0          LIT + B = MAR              % S(S)                       09270C00 D
0320  0100 0000 0000 00E0          SBASE = LIT                                             09280C00 D
0321  A809 0000 0C00 08F0          MR1; IF RDC                                             09290000 D
0322  AC08 0000 0C00 00F0          WHEN RDC THEN BEX                                       09300C00 D
0323  482D EC40 101C 00F0          A3 + B = A3,MAR2;JUMP                                   09310000 D
                                FETCHOPT0:                                                09320C00 D
0324  4809 A0C0 9030 00F0          A1 R = A3                                               09330C00 D
0325  1070 0000 0030 00A0          7 = LIT; 17 = SAR                                       0934CC00 D
0326  4809 E156 1C30 00F0          A3 AND LIT = A3            % B-FIELD                    09350C00 D
0327  4809 E140 003C 00F0          A3 + LIT = MAR             % S(B)                       C9360C00 D
0328  0100 00C3 003C 00A0          SBASE = LIT; 16 = SAR                                   09370C00 D
0329  A809 A0C1 1030 08F0          A1 L = A3; MR1; IF RDC                                  09380C00 D
032A  AC08 E000 9C30 00F0          WHEN RDC THEN A3 R = A3,BEX    % SY => A3               09390C00 D
032B  482D EC40 101C 00F0          A3 + B = A3,MAR2; JUMP     % SY + S(B)                  09400C00 D
                                FETCHOPT1:                                                09410C00 D
032C  4809 A000 9030 C0F0          A1 R = A3                                               0942C000 D
032D  1070 00C0 0000 C0A0          7 = LIT; 17 = SAR                                       09430C00 D
032E  4809 E156 1030 00F0          A3 AND LIT = A3            % B-FIELD                    09440C00 D
032F  4809 E140 0C3C 00F0          A3 + LIT = MAR             % B(B)                       09450000 D
0330  0080 0000 0000 00A0          BBASE = LIT; 16 = SAR                                   09460CC0 D
0331  A809 A001 0B00 08F0          A1 L = B; MR1; IF RDC                                   09470000 D
```

```
0332  AC08 0C40 8C80 00F0          WHEN RDC THEN B R = MIR,BEX    % SY => MIR          09480C00 D
0333  4809 E140 003C 00F0          A3 + LIT = MAR                 % S(B)               09490C00 D
0334  0100 0000 0000 00E0          SBASE = LIT                                         09500C00 D
0335  A809 0C40 1D00 08F0          B = A3,BMI; MR1; IF RDC        % B(B)=>A3,SY=>B      09510C00 D
0336  AC08 EC40 1C00 00F0          WHEN RDC THEN A3 + B = A3,BEX                       09520C00 D
0337  482D EC40 101C 00F0          A3 + B = A3,MAR2; JUMP         % SY+B(B)+S(B)       09530C00 D
                                INDIRECT:                                             09540C00 D
0338  4809 A000 9C00 00F0          A1 R = A3                      % OPCODE             09550C00 D
0339  2300 0000 0030 00B0          48=LIT;26=SAR                                       09560000 D
033A  4809 E15E 0000 00F0          A3 LSS LIT                                          0957CC00 D
033B  7019 0000 0030 00F0          IF TRUE THEN SKIP                                   0958CC00 D
033C  0B3C 0003 0000 C040          IFETCH2-1 = MPCR               % HALF-WORD INST     09590C00 D
                                INDIRECT1:                                            0960CC00 D
033D  3120 0000 0000 0060          FETNORM-1 = CPCR               % Y=Y+B(B)+S(S)      0961CC00 D
                                INDIRECT2:                                            09620C00 D
033E  A809 A0C3 C000 4CF0          A1 R = A1,CSAR; MR2; IF RDC    % Y => MAR2          09630C00 D
033F  8000 C000 0030 0020          20 = SAR                                            09640C00 D
0340  AC08 A001 4C30 C0F0          WHEN RDC THEN A1 L = A1,BEX                         0965CC00 D
0341  4809 0C40 0050 40F0          B = MIR, CSAR                  % (Y) = MIR          0966CC00 D
0342  4809 0C41 0B30 40FC          B L = B,CSAR                                        0967CC00 D
0343  4809 0C40 8B30 C0F0          B R = B                                             C968C00 D
0344  4809 AC5C 4C30 00F0          A1 OR B = A1                   % REPLACE A1(19-0)   09690C00 D
0345  4809 A000 9D00 00F0          A1 R = A3,BMI                  % (Y)=B              097C0C00 D
0346  0000 0000 0030 0020          16 = SAR                                            09710C00 D
0347  4809 E000 0030 C0F0          A3                                                  09720000 D
0348  5819 0000 0000 C0FC          IF LST THEN SKIP               % A1(16)=1?          09730C00 D
0349  0D3C 0000 0030 C040          INDIRECT3-1 = MPCR             % A1(16)=0           0974CC00 D
034A  4809 0C40 9030 C0F0          B R = A3                       % A1(16)=1           0975CC00 D
034B  9010 00C0 0030 00B0          ONE = LIT; 29 = SAR                                 09760CC0 D
034C  4809 E15B 0030 C0F0          A3 GTR LIT                                          0977CC00 D
034D  7019 0000 0030 00F0          IF FALSE THEN SKIP                                  09780C00 D
034E  33C0 0000 0030 0040          INDIRECT1-1 = MPCR                                  C9790000 D
034F  32B0 0C00 0C00 00C0          FETCHOPT1-1 = AMPCR                                 098C0000 D
0350  4809 E000 0030 C0F0          A3                                                  09810C00 D
0351  5833 00C0 0C00 00F0          IF LST THEN CALL ELSE SKIP                          09820C00 D
0352  481B 00C0 0000 C0F0          SKIP                                                C983CC00 D
0353  3230 0003 0000 C060          FETCHOPTO-1 = CPCR                                  C984CC00 D
0354  33D0 0000 0000 0040          INDIRECT2-1 = MPCR                                  0985CC00 D
                                INDIRECT3:                     % Y =>MAR2,(Y)=>B,MIR 09860C00 D
0355  4809 00C0 0038 00FC          LMAR                                                09870C00 D
0356  01A0 0003 0030 00E0          IAW=LIT                                             09880C00 D
0357  2300 0000 0030 C060          OUTPUT1-1 = CPCR               % SAVE (Y) = ICW     09890C00 D
0358  4809 00C0 0038 C0F0          LMAR                                                09900C00 D
0359  01C0 0000 C000 00E0          IAWADDR=LIT                                         0991CC00 D
035A  230C 0000 0030 0060          OUTPUT1-1 = CPCR               % SAVE Y = ADDR ICW  C9920C00 D
035B  1819 0003 0000 00F0          IF GC2 THEN SKIP               % REPEAT            09930C00 D
035C  0D40 0000 003C C040          INDIRECT4-1 = MPCR                                  09940C00 D
035D  3000 0000 0030 0060          S6YFETCH-1 = CPCR              % Y = Y+B(B)+S(6)    C995C000 D
035E  4809 A003 0038 00F0          A1 = MIR,LMAR                                       09960C00 D
035F  019C 0000 0030 C0E0          CINST=LIT                                           09970C00 D
0360  2300 0000 0030 0060          OUTPUT1-1 = CPCR               % SAVE C.I.          09980C00 D
0361  0B3C 0000 0030 0040          IFETCH2-1 = MPCR                                    09990000 D
                                INDIRECT4:                                            10000C00 D
0362  4809 0C40 9030 00F0          B R = A3                       % (Y)31-29           1001CC00 D
0363  9010 0003 0000 00B0          ONE = LIT; 29 = SAR                                 10020C00 D
0364  4809 E15B 0C0C C0F0          A3 GTR LIT                                          1003CC00 D
0365  7019 00C0 0000 00F0          IF FALSE THEN SKIP             % SKIP FOR OPTIONAL INDIRECTION 10040C00 D
0366  0D5C 0000 0030 0040          INDIRECT6-1 = MPCR             % NORMAL OR CA       10050C00 D
0367  4809 1800 8B00 00F0          B100 R = B                                          10060C00 D
0368  0000 00C0 0C00 0090          ZERO=LIT; 8=SAR                                     1007000C D
```

```
0369  4809 CC5C 2000 00F0          A2 OR B=A2                    %SET A2(23)                    1C08CC00 D
036A  3230 0000 0030 00C0          FETCHOPT0-1 = AMPCR                                          1C090000 D
036B  4809 E152 0C00 00F0          A3 EQL LIT                                                   1010CC00 D
036C  6833 0000 0000 C0F0          IF TRUE THEN CALL ELSE SKIP                                  1C110C00 D
036D  481B 0000 0000 00F0          SKIP                                                         1C120C00 D
036E  32B0 0000 0000 C060          FETCHOPT1-1 = CPCR                                           1013CC00 D
                                INDIRECT5:                                                      1C140C00 D
036F  4889 E0C0 0098 00F0          A3 = MIR,LMAR; SET GC2                                       1015CC00 D
0370  01B0 0003 0030 00E0          YADDR=LIT                                                    1C16CC00 D
0371  2300 0000 0030 0060          OUTPUT1-1 = CPCR              % SAVE Y AT 19                 1017CC00 C
0372  0B30 0000 0C30 0040          IFETCH2-1 = MPCR                                             1018CC00 D
                                INDIRECT6:                                                      10190000 D
0373  4809 E000 9030 C0F0          A3 R = A3                     % A3=(Y)31-29                  1C200C0C D
0374  1000 0000 0000 0000          1 = SAR                                                      1C210C00 D
0375  4809 E000 0000 00F0          A3                            % A3 = (Y)31,3C                 1022CC00 D
0376  580B 00C0 0C30 00F0          IF LST THEN STEP ELSE SKIP                                   1023CC00 D
0377  0D60 0000 0030 0040          INDIRECT8-1 = MPCR                                           1024CC00 D
0378  4809 18C0 8B00 C0FC          B130 R = B                    % NORMAL INDIRECTION            1C25CC00 D
0379  1000 0000 0000 0010          9=SAR                                                        1C26CC00 D
037A  4809 CC5C 20CC C0FC          A2 OR B=A2                    %SET A2(22)                    1027CC00 D
                                INDIRECT7:                                                      1C28C000 D
037B  3120 00C0 0000 C060          FETNORM-1 = CPCR                                             1029CC00 D
037C  4809 E000 0090 00F0          A3 = MIR                                                     1C30CC00 D
037D  36E0 0003 0030 0040          INDIRECT5-1 = MPCR                                           1C310C00 D
                                INDIRECT8:                                                      1032CC00 D
037E  4809 A0C0 90CC C0F0          A1 R = A3                     % CHECK CA                     1C330C00 D
037F  200C 0000 0030 003C          26 = SAR                      % OPCODE                       10340000, D
0380  0D70 0000 0000 0060          CONDCHECK-1 = CPCR                                           1035CC00 D
0381  3819 0000 0030 00F0          IF LC2 THEN SKIP                                             1036CC00 D
0382  0A5C 0000 003C 0040          IFETCH-1 = MPCR                                              10370000 D
0383  4809 2001 0B00 00F0          LIT L=B                       %SET BITS 22,23                1C38CC00 D
0384  2030 00C0 0090 0090          3=LIT; COMP 22=SAR                                           1039CC00 D
0385  4809 CC5C 2000 C0F0          A2 OR B=A2                    %SET A2(22,23)                 1040CC00 D
0386  37AC 0000 0000 0040          INDIRECT7-1 = MPCR                                           1C41CC00 D
                                ANCMP0:                         %A1=A-REG REFERENCED           1042CC00 D
0387  2920 0000 0030 0060          COUNTONES-1=CPCR              %A3=ONES COUNT                 1043CC00 D
0388  4809 A643 0090 00F0          A1*AMPCR=AMPCR                                               1C44C000 D
0389  0D8C 0000 0C3C 00C0          AOP-1=AMPCR                                                  1C45CC00 D
038A  4809 A0C0 003C C0F0          A1=MAR                        %ADDRESS OF A-REG              1046CC00 D
038B  A824 0000 0030 08F0          MR1;EXEC;IF RDC               %READ A-REG;EXECUTE CODE HANDLER 1C47CC00 D
038C  AC08 0000 0C0C C0F0          WHEN RDC THEN BEX             %B=(A)                         1C48CC00 D
038D  4836 0C40 0090 00F0          B;CALL                        %EXECUTE CODE HANDLER          1049CC00 D
038E  2CF0 00C0 0000 C040          GC2CHECK1-1=MPCR              %TERMINATE                     1050CC00 D
038F  2E50 0000 0030 C040          RPTCHECK1-1=MPCR              %NO TERMINATE                  1051CC00 D
                                %                                                               1C52CC00 D
                                $-CODE                                                          1053CC00 D
0390                            AOP:    AOP0-1=AMPCR. AOP1-1=AMPCR. AOP2-1=AMPCR. AOP3-1=AMPCR 1C54CC00 D
0394                                    AOP4-1=AMPCR. AOP5-1=AMPCR. AOP6-1=AMPCR. AOP7-1=AMPCR 1055CC00 D
                                $POP CODE                                                       1C56CC0C D
0398  4809 0C52 0000 C0F0          AOP0:   0 EQL B                                              1C57CC00 D
0399  602F 0000 0000 00F0                  IF FALSE THEN JUMP ELSE RETN                       1058CC00 D
039A  4809 0C52 0000 00F0          AOP1:   0 EQL B                                              1059CC0C D
039B  682F 0000 0000 C0F0                  IF TRUE THEN JUMP ELSE RETN                        1060CC0C D
039C  4839 0C52 000C 00F0          AOP2:   0 EQL B;IF MST THEN RETN                             1C61CC00 D
039D  682F 0000 0030 00FC                  IF TRUE THEN JUMP ELSE RETN                        1C62CC0C D
039E  4829 0C59 0090 00F0          AOP3:   0 GTR B;IF MST THEN JUMP                             1063CC0C D
039F  782F 0000 0030 00F0                  IF TRUE THEN JUMP ELSE RETN                        1064CCC0 D
03A0  483F 0000 0000 00F0          AOP4:   RETN                                                 1065CC00 D
03A1  4809 E000 0000 C0F0          AOP5:   A3.  IF NOT LST THEN JUMP ELSE RETN                  1066CC0C D
03A2  502F 0000 0030 00F0
```

```
03A3  4809 E00D 0030 00F0    AOP6:   A3.  IF LST THEN JUMP ELSE RETN                          10670000 0
03A4  582F 0000 0030 00F0
03A5  483F 0000 0000 00F0    AOP7:   RETN                                                    10680000 0
                             %                                                               10690000 0
                             ACMPO:   %CHECK CONDITION BITS FOR TERMINATING REPEAT           10700000 0
03A6  4809 A640 0040 00F0            A1+AMPCR=AMPCR              %A1=A-REG REFERENCED       10710000 0
03A7  0E10 0003 0030 00C0            AC0-1=AMPCR                                            10720000 0
03A8  9000 0000 0000 0030            COMP 1=SAR                                             10730000 0
03A9  4809 C001 9030 00F0            A2 C=A3                     %EOL IS LSB;GEO IS MSB;LIMITS 2ND MSB10740000 0
03AA  4824 0003 0030 00F0            EXEC                                                   10750000 0
03AB  4836 E000 0030 00F0            A3;CALL                     %CALL CONDITION HANDLER    10760000 0
03AC  2CF0 0000 0030 0040            GC2CHECK1-1=MPCR            %TERMINATE                 10770000 0
03A0  2E50 00C0 0030 0040            RPTCHECK1-1=MPCR                                       10780000 0
                             $-CODE                                                          10790000 0
03AE                         ACD:    AC00-1=AMPCR. ACD1-1=AMPCR. ACD2-1=AMPCR. ACD3-1=AMPCR 10800000 0
03B2                                 AC04-1=AMPCR. ACD5-1=AMPCR. AC06-1=AMPCR. AC07-1=AMPCR 10810000 0
                             $POP CODE                                                       10820000 0
03B6  502F 0000 0030 00F0    ACD0:   IF NOT LST THEN JUMP ELSE RETN                          10830000 0
03B7  582F 0000 0000 00F0    ACD1:   IF LST THEN JUMP ELSE RETN                              10840000 0
03B8  540F E000 0030 00FC    ACD2:   IF NOT LST THEN A3;STEP ELSE RETN                       10850000 0
03B9  482F 0000 0000 00F0            IF MST THEN JUMP ELSE RETN                             10860000 0
03BA  482F 0003 0000 00F0    ACD3:   IF MST THEN JUMP ELSE RETN                             10870000 0
03BB  402F 0000 0000 00FC    ACD4:   IF NOT MST THEN JUMP ELSE RETN                         10880000 0
03BC  540F E000 0030 00F0    ACD5:   IF NOT LST THEN A3;STEP ELSE RETN                       10890000 0
03BD  402F 0000 0030 00FC            IF NOT MST THEN JUMP ELSE RETN                         10900000 0
03BE  48C9 E00A 0030 00F0    AC06:   A3 0A0 0                    %TEST LIMITS BIT 2ND MSB   10910000 0
03BF  482F 0000 003C 00F0            IF MST THEN JUMP ELSE RETN                             10920000 0
03C0  4809 E00A 0000 00F0    AC07:   A3 0A0 0                    %TEST LIMITS BIT 2ND MSO   10930000 0
03C1  402F 00C3 0030 00F0            IF NOT MST THEN JUMP ELSE RETN                         10940000 0
                             %                                                               10950000 0
                             CONOCHECK:                          %RETURN LC1=>REPEATABLE    10960000 0
03C2  4809 0641 0010 24F0            AMPCR L=BR2;ASE             %LC2 => CHAR.ADDRESSABLE   10970000 0
03C3  0000 0000 0030 003C            COMP 8=SAR                  %SAVE RETURN ADDRESS IN BR2 10980000 0
03C4  4809 E640 0040 00F0            A3+AMPCR=AMPCR              %RETURN SAVED IN BR2;      10990000 0
03C5  0EAC 0000 0000 00C0            CONOTABLE-1=AMPCR           %SET CONDITION CODE        11000000 0
03C6  2809 0C41 8B30 00FC            B C=B;IF LC1                                           11010000 0
03C7  800C 00C0 003C 002C            20=SAR                                                 11020000 0
03C8  3824 0000 0030 00F0    SETUPCD:  EXEC;IF LC2                                          11030000 0
03C9  4809 2640 0C40 00F0            LIT+AMPCR=AMPCR             %SETUP TO EXECUTE CONDITIONS 11040000 0
03CA  0EB0 0000 0030 00C0            SETCOND-1=AMPCR                                        11050000 0
03CB  4809 2C48 0035 00F0            LIT NAN B=CTR               %SAVE SUBOPCODE IN CTR     11060000 0
03CC  0070 0000 0C30 0090            7=LIT;8=SAR                                            11070000 0
03CD  4820 0000 0030 00F0            JUMP                        %EXECUTE SETCOND + VALUE IN TABLE 11080000 0
03CE  0EC0 0000 0030 00C0    SETCOND:  SPECCOND-1=AMPCR          %0=LIT=> 02 OR 03 COPCODE  11090000 0
03CF  49C9 0000 0000 00F0            SET LC1                     %1=LIT R OR CA             11100000 0
0300  485B 0000 0030 00F0            SET LC2;SKIP                %2=LIT NOR P BUT CA        11110000 0
0301  49C9 0000 0030 00F0            SET LC1                     %3=LIT NOT CA BUT R        11120000 0
0302  4809 0F40 8040 00F0    CONDEXIT: BMAR R=AMPCR              %4=LIT NOT CA & NOT R      11130000 0
0303  4809 0000 0D30 00F0            BMI                         %RESET B                   11140000 0
03D4  4820 00C0 0030 00F0            JUMP                        %RETURN TO CALLING ROUTINE 11150000 0
                             SPECCOND:                                                      11160000 0
0305  4809 6640 0C40 00F0            CTR+AMPCR=AMPCR             %MUST BE 02/03             11170000 0
0306  0E00 00C0 0030 00C0            SUBTABLE-1=AMPCR                                       11180000 0
03D7  4809 E152 0030 00F0            A3 EQL LIT                  %CHECK FOR 03              11190000 0
0308  0030 0000 0030 00E0            3=LIT                                                  11200000 0
0309  6C0B 2640 0040 00FC            IF TRUE THEN LIT+AMPCR=AMPCR;STEP ELSE SKIP            11210000 0
030A  008C 00C0 0030 00EC            8=LIT                       %OFFSET PAST 02 CODES      11220000 0
03DB  4809 0000 0030 00F0            STEP                                                   11230000 0
03DC  3C70 0000 0030 C040            SETUPCD-1=MPCR              %USE CODE FOR REGULAR OPCODES 11240000 0
                             $-CODE                                                         11250000 0
```

```
                                    CONDTABLE: %4=NOT CA OR R;3=NOT CA/R;2=CA/NOT R;1=CA/R;0=C2 OR 03        11260C00 D
030D                                        4=LIT. 3=LIT. 0=LIT. 0=LIT. 4=LIT. 4=LIT. 4=LIT. 4=LIT. 1=LIT    11270C00 D
03E6                                        2=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT    11280C0C D
03EF                                        1=LIT. 1=LIT. 1=LIT. 2=LIT. 1=LIT. 1=LIT. 4=LIT. 4=LIT. 3=LIT    11290C00 D
C3F8                                        3=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT. 1=LIT. 3=LIT. 1=LIT    11300C00 D
0401                                        1=LIT. 1=LIT. 1=LIT. 1=LIT. 4=LIT. 4=LIT. 4=LIT. 4=LIT. 3=LIT    11310C00 D
040A                                        3=LIT. 3=LIT. 3=LIT. 4=LIT. 4=LIT. 4=LIT. 4=LIT. 4=LIT. 4=LIT    1132CC00 D
0413                                        4=LIT. 4=LIT. 4=LIT.  4=LIT. 4=LIT. 4=LIT. 4=LIT. 4=LIT. 4=LIT   11330000 D
041C                                        4=LIT                                                             11340C0C D
                                    SUBTABLE:    %USED FOR OPCODES C2 & 03                                    1135CC00 D
041D                                          1=LIT. 4=LIT. 4=LIT. 4=LIT. 1=LIT. 1=LIT. 4=LIT. 4=LIT          11360C00 D
0425                                        3=LIT. 3=LIT. 1=LIT. 3=LIT. 1=LIT. 1=LIT. 1=LIT. 3=LIT            1137C000 D
                                    $POP CODE                                                                     11380C00 D
                                    %                                                                             11390C00 D
                                    %                                                                             11400C00 D
                                    % ********************        LOGIC OPERATORS      ********************      11410C00 D
                                    OPR01:                                                                        11420000 D
042D  1270 00C0 0000 0060                   Y2FETCH-1=CPCR                                                    11430C00 D
042E  246C 0000 0C00 C060                   GETF-1=CPCR                 %RETURNS F2-FIELD IN A3               11440C00 D
042F  4809 E640 0040 00F0                   A3+AMPCR=AMPCR                                                    11450C00 D
0430  0400 0000 0000 C0C0                   OPR01XX-1=AMPCR                                                   1146CC00 D
0431  4809 A000 9000 00F0                   A1 R=A3                     %(A)=>B                               1147CC00 D
0432  B070 0000 0030 00A0                   23=SAR;7=LIT                                                      1148CC00 D
0433  4809 0F40 4C00 00F0                   BMAR=A1                     %SAVE ADDRESS OF Y                    11490C00 D
0434  4809 E156 0C0C 00F0                   A3 AND LIT=MAR                                                    11500C00 D
0435  A824 0C40 1C00 08F0                   B=A3;EXEC;MR1;IF RDC        %(Y)=>A3                          1151CC00 D
0436  AC28 0000 0C00 00F0                   WHEN RDC THEN BEX;JUMP      %(A)=B;ADDR OF Y=A1;(Y)=A3            11520C00 D
                                    %   ********************      OPCODE 02      **************************      11530C00 D
                                    OPR02:                                                                        11540C00 D
0437  127C 00C0 0030 0060                   Y2FETCH-1 = CPCR            % Y VALUE => B                         11550C00 D
0438  4809 0F40 0050 00F0                   BMAR = MIR                  % SAVE Y ADDR                          1156CC00 D
0439  2460 0003 0000 0060                   GETF-1=CPCR                 %RETURNS F2-FIELD IN A3               11570C00 D
043A  4809 E640 0040 00F0                   A3 + AMPCR = AMPCR                                                11580C00 D
043B  0480 0000 0030 C0C0                   OPR02XX-1 = AMPCR                                                 11590C00 D
C43C  4809 A000 9000 00F0                   A1 R = A3                                                         11600C00 D
043D  B070 0000 0030 00A0                   23=SAR;7=LIT                                                      1161CC00 D
043E  4824 E156 000C 00F0                   A3 AND LIT = MAR; EXEC      % A-REG                           11620C00 D
043F  A809 0000 0C00 08F0                   MR1; IF RDC                                                       11630C00 D
0440  AC28 0C40 1C00 00F0                   WHEN RDC THEN B=A3;BEX;JUMP     % A(A)=>B,Y=> A3                  11640C00 D
                                    % ********************        REPLACE LOGICALS       **************      11650000 D
                                    OPR03:                                                                        11660C00 D
0441  2460 00C3 003C 0060                   GETF-1=CPCR                 %RETURNS F2-FIELD IN A3               11670C00 D
0442  007C 0000 0000 00E0                   7=LIT                                                             11680C00 D
0443  4849 E152 003C 00F0                   A3 EQL LIT; SET LC2         %SIGNALS REPEAT INSTR.                11690000 D
0444  42C0 0000 0030 00C0                   OPR01-1=AMPCR                                                     11700C00 D
0445  67D9 E152 0C30 00F0                   IF FALSE THEN A3 EQL LIT;SET LC1;SKIP %F2=5=>SAME AS 3&RC1X7      11710C00 D
0446  0EE0 0003 0C0C 0040                   OPR03X7-1=MPCR                                                    11720C00 D
0447  0050 0000 0C30 C0E0                   5=LIT                                                             11730C00 D
0448  6C0D 2001 0B30 00F0                   IF TRUE THEN LIT L=B;STEP ELSE JUMP                               11740C0C D
0449  807C 00C0 0030 0090                   7=LIT;COMP 20=SAR           %SUBSTITUTE 7 IN F2 FIELD             11750C00 D
044A  482D AC5C 4030 00FC                   A1 OR B=A1;JUMP                                                   11760C00 D
                                    % ********************     DOUBLE WORD OPERATIONS        **************     11770C00 D
                                    OPR05:                                                                        11780C00 D
044B  1270 0003 0C30 0060                   Y2FETCH-1=CPCR              %MAR2=Y-ADDR;A3,MIR=(Y)               11790C00 D
044C  246C 0000 0030 C060                   GETF-1=CPCR                 %RETURNS F2-FIELD IN A3               11800C00 D
044D  4809 E640 0040 00F0                   A3+AMPCR=AMPCR                                                    1181C000 D
044E  050C 0000 0C3C 00C0                   OPR05XX-1=AMPCR                                                   11820C00 D
044F  4809 ACC0 9000 C0FC                   A1 R=A3                     %ISOLATE A-FIELD                      1183CC00 D
0450  807C 0000 0C30 C0A0                   7=LIT;23=SAR                                                      11840C00 D
0451  4824 E156 1030 00F0                   A3 AND LIT=A3;EXEC          %A3=A-FIELD                           11850C00 D
```

```
0452  482D 0C40 0030 00F0          B=MIR;JUMP                  %MAR2=Y-ADDR;MIR=(Y)                    1186CC00 D
                                % ************************    FLOATING POINT SUBROUTINES   ************1187CC00 D
                                OPR06:                                                                  1188CC00 D
0453  224C 0000 0030 0040          UNWRITTEN-1=MPCR            %PRINT ERROR AND RETURN TO IFETCH       1189CC00 D
                                %************************     EXTERNAL FUNCTION      ***************** 1190CC00 D
                                OPR07:                                                                  1191CC00 D
0454  127C 0000 003C C06C          Y2FETCH-1=CPCR                                                      1192CC00 D
0455  4809 0F40 0B30 00F0          BMAR=B                      %B=ADDRESS OF Y+(B)+(S)                 1193CC00 D
0456  246C 0000 0030 0060          GETF-1=CPCR                 %RETURNS F2-FIELD IN A3                 1194CC00 D
0457  48C9 E640 0040 C0F0          A3+AMPCR=AMPCR                                                      1195CC00 D
0458  0600 0C00 0030 00C0          OPR07XX-1=AMPCR             %SETUPT FOR JUM                         1196CC00 D
0459  4809 A0C0 9030 C0F0          A1 R=A3                     %ISOLATE A FIELD                        1197CC00 D
045A  B070 0003 003C 00A0          7=LIT;23=SAR                                                        1198CC00 D
045B  4824 E156 1030 00F0          A3 AND LIT=A3;EXEC                                                  1199CC0C D
045C  482D 0000 003C 00F0          JUMP                                                                1200CC00 D
                                % ************************    LOAD A     ***************************** 1201CC0C D
                                OPR10:                          % LA                                    1202CC00 D
045D  0CF0 0000 0030 C060          YFETCH-1 = CPCR              %PUTS PROPER INFO INTO MIR             1203CC00 D
045E  24C0 00C0 0030 0060          GETAA3-1=CPCR                %RETURNS A-FIELD IN A3&MAR             1204CC00 D
045F  22F0 0000 0030 C04C          OUTPUT-1=MPCR                                                       1205CC0C D
                                % ************************    LOAD A AND INDEX B     ***************** 1206CC00 D
                                OPR11:                          % LXB                                   1207CC00 D
0460  0CF0 0000 0030 0060          YFETCH-1 = CPCR              % PUTS INFO INTO MIR                    1208CC00 D
0461  24C0 0000 0030 0060          GETAA3-1=CPCR                %RETURNS A-FIELD IN A3&MAR             1209CC00 D
0462  9809 A000 9030 18F0          A1 R = A3;MW1;IF SAI         % WRITE A-REG                          1210CCC0 D
0463  1070 0000 C030 00A0          17 = SAR; 7 = LIT                                                   1211CC00 D
0464  4809 E156 1000 00F0          A3 AND LIT = A3              % P-FIELD = A3                         1212CC00 D
0465  9C08 E140 003C C0F0          WHEN SAI THEN A3+LIT=MAR                                            1213CC00 D
0466  0080 0000 0C00 C0A0          BBASE = LIT; 16 = SAR                                               1214CC00 D
0467  A809 0000 0030 08F0          MR1; IF RDC                                                         1215CC00 D
0468  AC08 0000 0C30 C0F0          WHEN RDC THEN BEX                                                   1216CC00 D
0469  01BC 0000 003C 00E0          YADDR=LIT                                                           1217CC0C D
046A  4809 0C46 0090 00F0          0 + B + 1 = MIR                                                     1218CC00 D
046B  22F0 0000 0C30 C040          OUTPUT-1=MPCR                                                       1219CC00 D
                                % ************************    LOAD DIFFERENCE     ******************** 1220CC00 D
                                OPR12:                            .%LDIF                            1221CC00 D
046C  0CFC 0000 003C 0060          YFETCH-1 = CPCR                                                     1222CC00 D
046D  24C0 00C0 0000 C060          GETAA3-1=CPCR                %RETURNS A-FIELD IN A3&MAR             1223CC00 D
046E  A809 0000 0030 08FC          MR1; IF RDC                                                         1224CC00 D
046F  AC08 E0C0 0C3C 00F0          WHEN RDC THEN A3 +1 = MAR,BEX     % MAR = A+1 REG                   1225CC00 D
0470  4809 0C40 1D3C 00F0          B = A3,BMI                                                          1226CC00 D
0471  4809 E0C0 0C30 00F0          A3 = MIR                                                            1227CC00 D
0472  4809 0C40 1D0C 00F0          B = A3,BMI                                                          1228CC00 D
0473  4809 EC5E 1090 00F0          A3 - B = A3,MIR                                                     1229CC00 D
0474  7409 E0DE 1090 00F0          IF NOT AOV THEN A3-1=A3,MIR                                         1230CC00 D
0475  2019 0000 0C30 C0F0          IF NOT LC1 THEN SKIP                                                1231CC00 D
0476  0EF0 0000 003C 0040          OPR36X1 -1 = MPCR                                                   1232CC00 D
0477  22FC 00C0 0030 0040          OUTPUT-1=MPCR                                                       1233CC00 D
                                % ************************    SUBTRACT A     ********************** 1234CC00 D
                                OPR13:                          % ANA                                   1235CC00 D
0478  0CF0 0000 003C C06C          YFETCH-1 = CPCR                                                     1236CCCC D
0479  24C0 0000 0000 0060          GETAA3-1=CPCR                %RETURNS A-FIELD IN A3&MAR             1237CC00 D
047A  A809 0000 0030 08F0          MR1; IF RDC                                                         1238CC00 D
047B  AC08 0000 0C30 00F0          WHEN RDC THEN BEX                                                   1239CC30 D
047C  4809 0C40 1D30 C0F0          B = A3,BMI                                                          1240CCCC D
047D  4809 EC5E 1090 C0F0          A3 - B = A3,MIR                                                     1241CC00 D
047E  7409 E0DE 1090 00F0          IF NOT AOV THEN A3-1=A3,MIR                                         1242CC00 D
047F  22F0 00C0 0030 0040          OUTPUT-1=MPCR                                                       1243CC00 D
                                % ************************    ADD A     ***************************** 1244CC00 D
                                OPR14:                          % AA                                    1245CCCC D
```

```
0480  0CF0 0000 0000 C060          YFETCH-1 = CPCR                                          12460C00 D
0481  2490 0000 0000 0060          GETAB-1=CPCR              %RETURNS A-FIELD IN B&MAR      12470C00 D
0482  A809 00C0 0000 C8F0          MR1; IF RDC                                              12480C00 D
0483  AC08 0000 0C90 00F0          WHEN RDC THEN BEX                                        1249CC00 D
0484  4809 EC40 0090 00F0          A3 + B = MIR                                             12500C00 D
0485  7C09 EC46 0090 00F0          IF AOV THEN A3+B+1=MIR                                   12510000 D
0486  6C09 00C0 0090 00F0          IF ABT THEN 0=MIR         %ELIMINATE NEGATIVE 0          12520C00 D
0487  22F0 00C0 0000 C040          OUTPUT-1=MPCR                                            12530C00 D
                                % ********************      LOAD SUM      ********************** 12540000 D
                                OPR15:                      % LSUM                         12550C00 D
0488  0CF0 0000 0C00 0060          YFETCH-1 = CPCR                                          12560C00 D
0489  24C0 0000 0090 0060          GETAA3-1=CPCR             %RETURNS A-FIELD IN A3&MAR     12570C00 D
048A  A809 00C0 0000 08F0          MR1; IF RDC                                              12580C00 D
048B  AC08 E0C0 1C0C C0F0          WHEN RDC THEN A3+1=A3,BEX %ADDRESS A+1 REGISTER          12590C00 D
048C  4809 E156 009C 00F0          A3 AND LIT=MAR            %A(8)=A(0)                     12600C00 D
048D  007C 00C0 000C 00E0          7=LIT                                                    12610000 D
048E  4809 0C40 1D9C 00F0          B = A3,BMI                                               12620C00 D
048F  4809 EC40 0C90 C0FC          A3 + B = MIR                                             12630C00 D
0490  7C09 EC46 0090 00F0          IF AOV THEN A3+B+1=MIR                                   12640C00 D
0491  6C09 0000 0C90 C0F0          IF ABT THEN 0=MIR         %ELIMINATE NEGATIVE 0          12650C00 D
0492  22F0 0000 0000 C040          OUTPUT-1=MPCR                                            12660C00 D
                                % ********************      LOAD NEGATIVE     ********************** 12670C00 D
                                OPR16:                      % LNA                          12680C00 D
0493  0CFC 0000 0000 0060          YFETCH-1 = CPCR                                          1269CC0C D
0494  249C 0000 0C00 0C60          GETAB-1=CPCR              %RETURNS A-FIELD IN B&MAR      12700000 D
0495  4809 E0C2 0090 00F0          NOT A3 = MIR                                             12710C00 D
0496  22F0 0000 009C C040          OUTPUT-1=MPCR                                            12720C00 D
                                % ********************      LOAD MAGNITUDE    ********************** 1273CC00 D
                                OPR17:                      % LM                           1274CC0C D
0497  0CF0 0000 0090 0060          YFETCH-1=CPCR                                            12750C00 D
0498  2490 0000 0090 0060          GETAB-1=CPCR              %RETURNS A-FIELD IN B&MAR      12760C00 D
0499  4809 E000 0C9C 00F0          A3                                                       12770C00 D
049A  4C09 E002 0C90 C0F0          IF MST THEN NOT A3 = MIR                                 12780C00 D
049B  22F0 0000 0000 C040          OUTPUT-1=MPCR                                            1279CC00 D
                                % ********************      LOAD B      ********************** 1280CC00 D
                                OPR20:                      % LB                           12810000 D
049C  0CF0 0000 0090 006C          YFETCH-1 = CPCR                                          12820C00 D
049D  24CC 0000 0090 0060          GETAA3-1=CPCR             %RETURNS A-FIELD IN A3&MAR     1283CC00 D
049E  4809 E002 0000 00F0          NOT A3                                                   12840C00 B
049F  0A5C 0000 0C00 00C0          IFETCH-1=AMPCR                                           1285CC00 D
04A0  6829 0000 0000 00F0          IF ABT THEN JUMP          % ABT,A-FIELD = 0              12860000 D
04A1  4809 E140 009C C0F0          A3 + LIT = MAR                                           12870C00 D
C4A2  0080 0000 0C0C 00E0          BBASE = LIT                                              12880C00 D
C4A3  22F0 0000 0090 0040          OUTPUT-1=MPCR                                            1289CC00 D
                                % ********************      ADD B       ********************** 12900C00 D
                                OPR21:                      % AB                           1291C000 D
04A4  0CFC 0000 000C C060          YFETCH-1 = CPCR                                          1292CC00 D
04A5  24C0 00C0 009C 0060          GETAA3-1=CPCR             %RETURNS A-FIELD IN A3&MAR     12930C00 D
04A6  4809 E002 0000 00F0          NOT A3                                                   1294CC00 D
04A7  0A50 0000 0000 C0C0          IFETCH-1=AMPCR                                           1295CC00 D
C4A8  6829 0000 0090 00F0          IF ABT THEN JUMP          % ABT, A-FIELD = 0             12960C00 D
04A9  4809 E140 0D9C 00F0          A3 + LIT = MAR,BMI                                       12970C00 D
04AA  008C 0000 0C00 C0A0          BBASE = LIT; 16 = SAR                                    12980C00 D
04AB  A809 0000 0000 08F0          MR1; IF RDC                                              12990C00 D
04AC  AC08 0C40 1C0C 00F0          WHEN RDC THEN B = A3,BEX                                 1300CC00 D
04AD  4809 EC40 0C90 C0F0          A3+B=MIR                  %(Y)+B(A)=>B(A)                13010000 D
04AE  7C09 EC46 0C90 C0F0          IF AOV THEN A3+B+1=MIR                                   1302CC00 D
04AF  6C09 0000 C090 00F0          IF ABT THEN 0=MIR         %ELIMINATE NEGATIVE 0          1303CC00 D
04B0  22FC 0000 0090 0040          OUTPUT-1=MPCR                                            13040C00 D
                                % ********************      SUBTRACT B  ********************** 13050C00 D
```

```
                                OPR22:                                  % ANB                                   13060C00 D
04B1  0CF0 C000 C000 0060               YFETCH-1 = CPCR                                                         13070C00 D
04B2  24CC 0000 0000 0060               GETAA3-1=CPCR                   %RETURNS A-FIELD IN A3&MAR              13080C00 D
04B3  4809 E002 0000 C0F0               NOT A3                                                                  1309CC0C D
04B4  0A50 00C0 0030 C0CC               IFETCH-1=AMPCR                                                          1310CC00 D
04B5  6829 0000 0000 00F0               IF ABT THEN JUMP                %A- FIELD = 0,ABT                       13110000 D
0496  4809 E140 003C 00F0               A3 + LIT = MAR                                                          1312CC00 D
04B7  0080 0000 0000 00A0               BBASE = LIT; 16 = SAR                                                   13130C00 D
04B8  A809 0000 0030 08F0               MR1; IF RDC                                                             13140C00 D
04B9  AC08 0000 0C00 C0F0               WHEN RDC THEN BEX               % READ B(A)                             13150C00 D
04BA  4809 0C40 1D30 C0F0               B=A3,BMI                                                                1316CC00 D
04BB  4809 EC5E 0090 00F0               A3-B=MIR                                                                13170C00 D
04BC  7409 EC58 0090 C0F0               IF NOT AOV THEN A3-B-1=MIR                                              13180C00 D
04BD  22FC 0000 0000 0040               OUTPUT-1=MPCR                                                           1319CC00 D
                                %  ********************       STORE B      ********************                13200C00 D
                                OPR23:                                  % SB                                    1321CC00 D
04BE  4809 E000 0090 00F0               A3 = MIR                                                                1322CC00 D
04BF  249C 00C0 0000 0060               GETAB-1=CPCR                    %RETURNS A-FIELD IN B&MAR               13230C00 D
04C0  4809 0C42 0000 00F0               NOT B                                                                   13240C00 D
04C1  6C0B 0000 0090 00F0               IF ABT THEN BC00 = MIR ELSE SKIP                                        1325CC00 D
04C2  0F0C 0000 003C 0040               AZERO-1 = MPCR                                                          13260C00 D
04C3  4809 2C40 003C C0F0               LIT + B = MAR                                                           1327C000 D
04C4  008C 000D 0030 00A0               BBASE=LIT;16=SAR                                                        13280C00 D
04C5  A809 00C0 003C 08F0               MR1; IF RDC                                                             13290000 D
04C6  AC08 0000 0C30 C0F0               WHEN RDC THEN BEX                                                       133CC000 D
04C7  4809 0C41 0B30 00F0               B L = B                                                                 1331CC00 D
04C8  4809 0C40 8090 00F0               B R = MIR                       % B(A) 15-C = MIR                       13320C00 D
04C9  4809 E000 003C C0F0       AZERO:  A3 = MAR                                                                13330C00 D
04CA  1550 0000 0C0C C04C               YSTORE-1 = MPCR                                                         13340C00 D
                                %  ********************       STORE  A      ****************************        13350C00 D
                                OPR24:                                  % SA                                    1336CC00 D
04CB  2490 0000 0C30 C060               GETAB-1=CPCR                    %RETURNS A-FIELD IN B&MAR               1337CC00 D
04CC  A809 00C0 0000 08F0               MR1; IF RDC                     % READ A(A)                             13380C00 D
04CD  AC08 0000 0C30 C0F0               WHEN RDC THEN BEX                                                       13390C00 D
04CE  48C9 0C40 0090 C0F0               B = MIR                         % STORE A(A)                            13400C00 D
04CF  1550 0000 0030 D04C               YSTORE - 1 = MPCR                                                       1341CC0C D
                                %  ********************       STORE A AND INDEX B      ********************    1342CC0C D
                                OPR25:                                  % SXB                                   13430C00 D
04D0  A809 0000 0030 C8F0               MR1; IF RDC                                                             13440C00 D
04D1  AC08 0000 0C00 C0F0               WHEN RDC THEN BEX                                                       13450C00 D
04D2  4809 0C46 0090 C0F0               B+1=MIR                         %INCREMENT B(B)                         13460C00 D
04D3  3120 000D 0030 C060               FETNORM-1 = CPCR                % A3,MAR2 = Y ADDR                      13470C00 D
04D4  4809 A0C0 8B0C C0F0               A1 R = B                                                                13480C00 D
04D5  1070 00C0 0000 C0AC               7 = LIT; 17 = SAR                                                       13490C00 D
04D6  4809 2C56 0B0C C0F0               LIT AND B = B                                                           13500C00 D
04D7  4809 2C40 003C 00FC               LIT + B = MAR                   % ADDR B(B)=>MAR                        13510C00 D
04D8  0080 000D 0030 00EC               BBASE = LIT                                                             1352CC00 D
04D9  9809 A0C0 8B00 18F0               A1 R = B; MW1; IF SAI           % INC B(B)                              1353CC00 D
04DA  B070 00C0 0030 C0A0               23 = SAR; 7 = LIT                                                       1354CC00 D
04DB  9C08 2C56 000C 00F0               WHEN SAI THEN LIT AND B = MAR   % A-REG                                 13550C00 D
04DC  A809 E00D 0090 08FC               A3 = MIR; MR1; IF RDC           % Y ADDR=> MIR                          13560C00 D
04DD  AC08 A000 9C3C 00F0               WHEN RDC THEN A1 R = A3,BEX     % A(A)=>B                               13570C00 D
04DE  8000 0000 0030 0020               20 = SAR                                                                13580C00 D
04DF  4809 E156 1C3C 00FC               A3 AND LIT = A3                                                         13590C00 D
04E0  4809 0C40 0D90 00FC               B = MIR, BMI                    % Y ADDR=>E, A(A)=>MIR                  13600C00 D
04E1  4809 E640 0090 00F0               A3 + AMPCR = AMPCR                                                      13610C00 D
04E2  1B60 00C0 0030 00C0               KSTABLE-1 = AMPCR                                                       13620C00 D
04E3  4809 0C43 001C 00F0               B = MAR2                                                                13630C00 D
04E4  A824 00C0 0030 CCFC               EXEC; MR2; IF RDC                                                       13640C00 D
04E5  482D 0000 0000 00FC               JUMP                                                                    1365CC00 D
```

```
                                        %   ********************       STORE NEGATIVE       ********************  13660000 D
                                        OPR26:                                                                   13670000 D
04E6  249C 0000 0000 0060                       GETAB-1=CPCR            % SNA                                    13680000 D
04E7  A809 0000 0000 08F0                       MR1; IF RDC             %RETURNS A-FIELD IN B&MAR                13690000 D
04E8  AC08 00C0 0C30 00F0                       WHEN RDC THEN BEX                                                13700000 D
04E9  4809 0C42 0030 C0F0                       NOT B = MIR             % B => A(A)                              13710000 D
04EA  155C 00C0 0000 0040                       YSTORE-1 = MPCR         % ONE:S COMP                             13720000 D
                                        %   ********************     STORE MAGNITUDE     ************************  13730000 D
                                        OPR27:                                                                   13740000 D
04E8  2490 0000 003C 0060                       GETAB-1=CPCR            % SM                                     13750000 D
04EC  A809 0000 0000 08FC                       MR1; IF RDC             %RETURNS A-FIELD IN B&MAR                13760000 D
04ED  AC08 0000 0C30 00FC                       WHEN RDC THEN BEX                                                13770000 D
04EE  4809 0C40 0090 00F0                       B = MIR                 % A(A)=> B                               13780000 D
04EF  4C09 0C42 0090 C0F0                       IF MST THEN NOT B = MIR % COMP B IF BIT 31 = 1                   13790000 D
C4F0  4809 E000 003C C0F0                       A3 = MAR                % RESTORE TO READ B(B)                    13800000 D
04F1  1550 0000 0000 0040                       YSTORE-1 = MPCR                                                  13810000 D
                                        %    ********************     CLEAR BIT     ****************************  13820000 D
                                        OPR32:                                  % BZ                             13830000 D
                                        %     TAKE THE NUMBER IN AK FIELD, CLEAR THAT BIT IN Y                   13840000 D
04F2  1270 00C0 0000 0060               Y2FETCH-1=CPCR          %GET Y-ADDRESS AND IGNORE K                      13850000 D
04F3  4809 A0C1 1030 C0F0               A1 L=A3                 %ISOLATE A-FIELD                                 13860000 D
04F4  2000 00C0 003C 0030               COMP 6=SAR                                                               13870000 D
04F5  4809 E0C0 9C30 00F0               A3 R=A3                                                                  13880000 D
04F6  2000 0000 0000 0030               26=SAR                                                                   13890000 D
04F7  4809 E000 0030 80F0               A3=SAR                  %SAR=AK-FIELD                                    13900000 D
04F8  4809 0C41 9030 40F0               B C=A3,CSAR             %BIT(AK) IN LSB                                  13910000 D
04F9  2C19 E0DD 8080 C0F0               IF LC1 THEN A3 OR 8001 C=MIR;SKIP                                        13920000 D
04FA  4809 E0CF 8C80 00F0               A3 AND B110 C=MIR       %CLEAR BIT IF OPR32 ELSE SET                     13930000 D
C4FB  2370 00C0 0000 0040               YOUT-1=MPCR                                                              13940000 D
                                        %    ********************      SET BIT      ****************************  13950000 D
                                        OPR33:                                  % BS                             13960000 D
                                        %     SAME CODE AS OPCODE 32 SO WILL USE THAT EXCEPT SET THE BIT         13970000 D
04FC  49C9 0000 0030 C0F0               SET LC1                                                                  13980000 D
04FD  4F1C 0000 0000 0040               OPR32-1 = MPCR                                                           13990000 D
                                        %   ********************     REPLACE ADD     ************************   14000000 D
                                        OPR34:                                  % RA                             14010000 D
04FE  0CFC 0000 0030 C060               YFETCH-1 = CPCR         % Y VALUE => A3,MIR                               14020000 D
04FF  24C0 00C0 0030 0060               GETAA3-1=CPCR           %RETURNS A-FIELD IN A3&MAR                       14030000 D
0500  A809 E0C0 1000 08F0               A3 + 1 = A3;MR1; IF RDC                                                  14040000 D
0501  AC08 E156 0C0C C0F0               WHEN RDC THEN A3 AND LIT=MAR,BEX                                         14050000 D
0502  4849 0C40 1D3C 00F0               B=A3,BMI;SET LC2        %A(A)=>A3,Y=>B;SIGNAL REPEAT                     14060000 D
0503  4809 EC40 1090 C0F0               A3 + B = A3,MIR                                                          14070000 D
0504  7C09 E0C0 1090 00F0               IF AOV THEN A3+1=A3,MIR                                                  14080000 D
0505  6C09 0000 1090 C0F0               IF ABT THEN 0=A3,MIR                                                     14090000 D
0506  2300 0000 0030 0060               OUTPUT1-1 = CPCR        % A(A)+Y=>A(A+1)                                 14100000 D
0507  1550 0000 0000 0040               YSTORE - 1 = MPCR                                                        14110000 D
                                        %   ********************     REPLACE INCREMENT     ********************   14120000 D
                                        OPR35:                                  % RI                             14130000 D
0508  0CFC 0000 0C00 0060               YFETCH-1 = CPCR         % Y VALUE => A3,MIR                               14140000 D
0509  4849 E0C0 0C90 00F0               A3+1=MIR;SET LC2        %(Y)+1=>MIR;SIGNAL REPEAT                        14150000 D
050A  7C09 E0C6 0C90 00F0               IF AOV THEN A3+1+1=MIR                                                   14160000 D
050B  6C09 0000 0090 00FC               IF ABT THEN 0=MIR       %ELIMINATE NEGATIVE 0                            14170000 D
050C  2490 00C0 0000 006C               GETAB-1=CPCR            %RETURNS A-FIELD IN B&MAR                        14180000 D
050D  2300 00C0 0C30 0060               OUTPUT1-1 = CPCR        % Y+1 => A(A)                                    14190000 D
050E  1550 0000 0030 0040               YSTORE-1=MPCR                                                            14200000 D
                                        %   ********************     REPLACE SUBTRACT     ********************   14210000 D
                                        OPR36:                                  % RAN                            14220000 D
050F  49C9 0000 0030 C0F0               SET LC1                                                                  14230000 D
0510  46BC 0000 0030 0040               OPR12 - 1 = MPCR        % SAME CODE THRU HERE                            14240000 D
0511  2300 C000 0000 006C       OPR36X1:  OUTPUT1-1 = CPCR                                                       14250000 D
```

```
0512  4849 00C0 0000 C0F0          SET LC2                    %SIGNALS REPEAT                  14260C00 D
0513  1550 0000 0030 C040          YSTORE - 1 = MPCR                                           14270C00 D
                                 %  ********************     REPLACE DECREMENT      **********************  14280C00 D
                                 OPR37:                                %RD                    14290C00 D
0514  0CF0 00C0 0030 C060          YFETCH-1=CPCR                                               14300C00 D
0515  2490 0000 0030 C060          GETAB-1=CPCR               %RETURNS A-FIELD IN B&MAR        14310C00 D
0516  4809 E0DE 0090 00F0          A3-1=MIR                                                    14320C00 D
0517  7409 E0D8 0C90 C0F0          IF NOT AOV THEN A3-1-1=MIR                                  14330C00 D
0518  2300 0000 0030 0060          OUTPUT1 - 1 = CPCR         % A(A)-1 => A(A)                 14340C00 D
0519  1550 00C0 0030 0040          YSTORE - 1 = MPCR                                           14350000 D
                                 % ************************   MULTIPLY A(A)XY = A(A+1),A(A)    14360C00 D
                                 OPR40:     %A3=(Y)=MULTIPLIER                               14370C00 D
051A  0CF0 0000 0000 C060          YFETCH-1=CPCR              %A3=(Y)                          14380C00 D
051B  2270 0000 0030 0060          PUTPAR-1=CPCR              %SAVE A2                         14390C00 D
051C  2490 00C0 0000 0060          GETAB-1=CPCR               %B,MAR=A-FIELD                   14400C00 D
051D  A809 E0C0 2C30 08F0          A3=A2;MR1;IF RDC           %A2=(Y)                          14410C00 D
051E  AC08 0000 0C30 C0FC          WHEN RDC THEN BEX          %B=(A(A))                        14420C00 D
051F  2980 0000 0C30 0060          MULTIPLY-1=CPCR            %A1=MOST;A3=LEAST                14430C00 D
0520  4809 E000 0C90 00F0          A3=MIR                     %OUTPUT LEAST                    14440C00 D
0521  9809 00C0 0000 18F0          MW1;IF SAI                                                  14450C00 D
0522  9C08 0F46 0C0C 00F0          WHEN SAI THEN BMAR+1=MAR                                    14460C00 D
0523  4809 AC00 0090 00F0          A1=MIR                     %OUTPUT MOST                     14470C00 D
0524  9809 0000 0030 18FC          MW1;IF SAI                                                  14480C00 D
0525  9C08 0000 0030 00F0          WHEN SAI THEN 0;STEP                                        14490C00 D
0526  0A5C 0C00 0030 00C0          IFETCH-1=AMPCR                                              14500000 D
0527  22AC 0000 0000 0040          GETPAR-1=MPCR              %RESTORE A2;RETURN TO IFETCH     14510C00 D
                                 % ************************   DIVIDE A(A+1) & A(A) / Y=>A(A+1) + A(A)  14520C00 D
                                 OPR41:  %DIVIDE A1 & A3 / B =A1(QUOTIENT) + A3(REMAINDER     14530C00 D
0528  0CF0 0000 0030 0060          YFETCH-1=CPCR              %MIR=DIVISOR                     14540C00 D
0529  2270 0000 0030 C060          PUTPAR-1=CPCR              %SAVE A2                         14550C00 D
052A  4809 E0C0 0090 00F0          A3=MIR                     %MIT=DIVISOR=(Y)                 14560C00 D
052B  24C0 0000 0000 0060          GETAA3-1=CPCR                                               14570C00 D
052C  4809 E0C0 000C 00F0          A3+1=MAR                   %A3=A ADDRESS                    14580C00 D
052D  4809 2F56 0C0C 00F0          LIT AND BMAR=MAR           %CONVERT A(8) TO A(0)            14590C00 D
052E  A809 0C00 C030 08F0          MR1;IF RDC                 %READ A+1                        14600C00 D
052F  AC08 E000 0C9C 00F0          WHEN RDC THEN A3=MAR,BEX   %MAR SETUP FOR LEAST DIVIDEND    14610C00 D
0530  4809 0C40 1D9C 00F0          B=A3,BMI                   %A3=DIVIDEND;B=DIVISOR           14620C00 D
0531  2A8C 0C00 0030 0060          DIVIDE-1=CPCR              %RETURNS A2=QUOTIENT;A1=REMAINDER  14630C00 D
0532  4809 C0C0 009C 00F0          A2=MIR                     %SETUP TO OUTPUT QUOTIENT        14640000 D
0533  6C09 0000 C090 C0F0          IF ABT THEN 0=MIR                                           14650C00 D
0534  9809 0000 0030 18F0          MW1;IF SAI                 %OUTPUT QUOTIENT                 14660C30 D
0535  9C08 0F46 000C 00F0          WHEN SAI THEN BMAR+1=MAR                                    14670C00 D
0536  4809 2F56 00DC C0F0          LIT AND BMAR=MAR           %CONVERT A(8) TO A(0)            14680C00 D
0537  0070 0000 0000 00E0          7=LIT                                                       14690C00 D
0538  4809 A000 0C90 00F0          A1=MIR                     %OUTPUT REMAINDER                14700C00 D
0539  6C09 A002 0090 00F0          IF ABT THEN NOT A1=MIR                                      14710C00 D
053A  9809 0000 0C30 18F0          MW1;IF SAI                                                  14720C00 D
053B  9C08 0000 0C00 00F0          WHEN SAI THEN 0;STEP                                        14730C00 D
053C  22A0 0000 0030 0060          GETPAR-1=CPCR              %RESTORE A2                      14740C0C D
053D  4809 C001 A030 40F0          A2 C=A2,CSAR                                                14750C00 D
053E  900C 0000 0030 0030          COMP 3=SAR                                                  14760C00 D
053F  2C19 D81D AC30 C0FC          IF LC1 THEN A2 OR B100 C=A2;SKIP                            14770C00 D
0540  4809 D80F A090 00F0          A2 AND B011 C=A2                                            14780C00 D
0541  0A5C 00C0 0C00 0040          IFETCH-1=MPCR                                               14790C00 D
                                 % ************************   COMPARE BIT TO ZERO      **********  14800C00 D
                                 OPR42:                                %BC                    14810C00 D
0542  1270 0000 0C00 0060          Y2FETCH-1=CPCR             %GET (Y) INTO E                  14820C00 D
0543  4809 0C40 1C00 00F0          B=A3                       %CHECK BIT IN A3=(Y)             14830000 D
0544  4809 A0C0 8B0C C0F0          A1 R=B                                                      14840C00 D
0545  83F0 00C0 0030 00A0          63=LIT;20=SAR                                               14850C00 D
```

```
0546  4809 2C56 0000 80FC          LIT AND B=SAR              %AK FIELD IS SHIFT AMOUNT                  1486CC00 D
0547  4809 E0C0 9000 C0F0          A3 R=A3                    %BIT TO COMPARE IS LS                      1487CC00 D
0548  4849 E00D 0C00 00F0          A3;SET LC2                 %CHECK LSB; SIGNAL REPEAT WITH LC2         1488CC00 D
0549  5C19 D8CE 200C 00F0          IF LST THEN A2 AND B011 = A2;SKIP                                     1489CC00 D
054A  4809 D81C 2030 00F0          A2 OR B10C=A2              %SET EQUAL CONDITION BIT IN ASR            1490CC0C D
054B  0A50 0000 0C3C C040          IFETCH-1=MPCR                                                         1491CC00 D
                                % ********************      COMPARE INDEX INCREMENT       **********    1492CC00 D
                                OPR43:                     %CXI                                       1493CC00 D
054C  0CFC 0000 0030 0060          YFETCH-1=CPCR           %Y VALUE =A3,MIR                           1494CC00 D
054D  249C 0000 0000 0060          GETAB-1=CPCR            %RETURNS A-FIELD IN B&MAR                  1495CC00 D
054E  4809 0C52 C000 00F0          0 EQL B                 %TEST FOR B(0) => NOOP IF TRUE             1496CC00 D
054F  6019 00C0 0000 00FC          IF FALSE THEN SKIP                                                    1497CC00 D
0550  0A50 0000 0030 C040          IFETCH-1=MPCR                                                         1498CC00 D
0551  4809 2C40 0C3C 00F0          LIT+B=MAR               %B(A) ADDRESS                              1499CC00 D
0552  0080 0000 0C00 00A0          BBASE=LIT;16=SAR                                                      1500CC00 D
0553  A809 C0C1 AC30 08F0          A2 C=A2;MR1;IF RDC      %GET LIMITS BIT TO MSB                     1501CC00 D
0554  A000 00C0 0030 0030          COMP 2=SAR                                                            1502CC00 D
0555  AC08 00C0 0C00 00F0          WHEN RDC THEN BEX       %READ B(A)                                 1503CC00 D
0556  4809 0C40 1D30 40F0          B=A3,BMI,CSAR           %A3=B(A);B=(A)                             1504CC00 D
0557  4809 E0C0 CC30 C0F0          A3+1=MIR                %INCREMENT B(A) THEN TEST B(A)>A(A)        1505CC00 D
0558  4849 EC4C 0000 00F0          A3 XOR B;SET LC2        %TEST FOR UNLIKE SIGNS;SIGNAL REPEAT       1506CC00 D
0559  480B EC5E 0000 C0F0          A3 GEQ B; IF MST THEN STEP ELSE SKIP                                  1507CC00 D
055A  77DB 00C0 0C30 00F0          SKIP; IF FALSE THEN 0=MIR;SET LC1  %SET B(A)=0                        1508CC00 D
055B  7FC9 00C0 009C 00F0          IF TRUE THEN 0=MIR;SET LC1 %SET B(A) = 0                              1509CC00 D
055C  2C19 D81D AC00 C0F0          IF LC1 THEN A2 OR B100 C=A2;SKIP  %SET OUT OF LIMITS                  1510CC00 D
055D  4809 D80F A000 00F0          A2 AND B011 C=A2        %CLEAR LIMITS BIT                          1511CC00 D
055E  22FC 00C0 0030 0040          OUTPUT-1=MPCR                                                         1512CC00 D
                                % ********************       COMPARE      **************************** 1513CC00 D
                                OPR44:                     %C                                         1514CC00 D
055F  0CF0 0000 0030 0060          YFETCH-1=CPCR           %Y VALUE = A3,MIR                          1515CC00 D
0560  249C 0000 0000 0060          GETAB-1=CPCR            %RETURNS A-FIELD IN B&MAR                  1516CC00 D
0561  49C9 E000 0030 00F0          A3=MIR;SET LC1                                                        1517CC00 D
0562  A809 0003 0C00 08F0          MR1; IF RDC                                                           1518CC00 D
0563  AC08 0C00 0C0C 00F0          WHEN RDC THEN BEX       %A(A)                                      1519CCC0 D
0564  4809 0C40 1D3C C0F0          B=A3,BMI                %A(A)=A3;Y=>B                              1520CC00 D
0565  286C 00C0 0030 0040          SETCOMPARE-1=MPCR       %A3=A(A);B=(Y)                             1521CC0C D
                                %   **************      COMPARE LIMITS      ************************** 1522CC00 D
                                OPR45:                     % CL                                       1523CC00 D
                                %   IF Y BETWEEN A(A) AND A(A+1) OR = A(A) THEN CD WITHIN LIMITS      1524CC00 D
                                %   ELSE OUTSIDE OF LIMITS                                            1525CC00 D
0566  0CF0 00C0 003C 0060          YFETCH-1 = CPCR         % Y VALUE => A3,MIR                         1526CC00 D
0567  2490 0000 0030 0060          GETAB-1=CPCR            %RETURNS A-FIELD IN B&MAR                  1527CC00 D
0568  A809 0000 0030 08F0          MR1;IF RDC                                                            1528CC00 D
0569  0A50 00C0 0C3C C0C0          IFETCH-1=AMPCR          %USE IDENTICAL CODE IN OPR74X5             1529CC00 D
056A  AC08 0F46 0C3C 00F0          WHEN RDC THEN BMAR+1=MAR,BEX                                          1530CC00 D
056B  4809 2F56 000C 00F0          LIT AND BMAR=MAR        %CONVERT A(8) TO A(0)                      1531CC00 D
056C  A809 0C40 4030 C8F0          B=A1;MR1;IF RDC         %A1=A(A)                                   1532CC00 D
056D  AFC8 0C00 CC30 00F0          WHEN RDC THEN BEX;SET LC1                                             1533CC00 D
056E  4849 0C40 1C30 00F0          B=A3;SET LC2            %%A3=A(A+1);SIGNAL REPEAT                  1534CC00 D
056F  4809 A000 0D30 00F0          A1=MIR,BMI              %MIR=A(A);B=(Y)                            1535CC00 D
0570  0F1C 0000 0030 0040          OPR74XX5-1=MPCR                                                       1536CC00 D
                                %   ******************      COMPARE MASKED      ********************** 1537CC00 D
                                OPR46:                     % CM                                       1538CC00 D
0571  0CFC 0000 0030 0060          YFETCH-1 = CPCR         % Y VALUE => A3,MIR                         1539CC00 D
0572  249C 0000 0030 C060          GETAB-1=CPCR            %RETURNS A-FIELD IN B&MAR                  1540CC00 D
0573  A809 0000 C030 C8F0          MR1; IF RDC                                                           1541CC00 D
0574  AFC8 0F46 0C3C 00F0          WHEN RDC THEN BMAR+1=MAR,BEX;SET LC1                                  1542CC00 D
0575  4809 2F56 000C C0F0          LIT AND BMAR=MAR        %CONVERT A(8) TO A(0)                      1543CCCC D
0576  A809 EC56 1030 08FC          A3 AND B = A3,MIR;MR1; IF RDC                                         1544CC00 D
0577  AC08 0000 0C00 00F0          WHEN RDC THEN BEX       % A(A+1) => B                              1545CC00 D
```

```
0578  4809 0C40 1D30 00FC          B = A3,BM1                    % A3=(Y+1),B=(A)AND(Y)                  15460C00 D
0579  2860 0000 0000 0040          SETCOMPARE-1=MPCR             %A3=A(A) AND Y; B=A(A+1)               15470C00 D
                                %  ********************     COMPARE GATED     **************************  15480C00 D
                                OPR47:                                                                  15490C00 D
057A  0CF0 0000 0030 006C          YFETCH-1 = CPCR               % Y VALUE => A3,MIR                     15500C00 D
057B  2490 0000 0030 0060          GETAB-1=CPCR                  %RETURNS A-FIELD IN B&MAR              15510C00 D
057C  A809 0000 0030 08F0          MR1; IF RDC                                                          15520C00 D
057D  AFC8 0F46 0C0C C0F0          WHEN RDC THEN BMAR+1=MAR,DEX;SET LC1                                 15530C00 D
057E  4809 2F56 000C 00F0          LIT AND BMAR=MAR              %CONVERT A(8) TO A(0)                  15540C00 D
057F  7409 EC58 1B30 00F0          IF NOT AOV THEN A3-B-1=A3,B                                          15550C00 D
0580  A809 EC5E 1B30 08F0          A3-B=A3,B;MR1;IF RDC          %TAKE DIFFERENCE                       15560C00 D
0581  7409 E0DE 1B30 C0F0          IF NOT AOV THEN A3-1=A3,B                                            15570C00 D
0582  4C09 0C42 1000 C0F0          IF MST THEN NOT B=A3          %PUT ABSOLUTE VALUE IN A3              15580C00 D
0583  AC08 00C0 0C30 00F0          WHEN RDC THEN BEX             %READ A(A+1)                           15590C00 D
0584  2860 0000 000C 0040          SETCOMPARE-1=MPCR             %A3=ABS(Y-A(A)); B=A(A+1)              15600C00 D
                                % ************************     JUMP PARITY / DOUBLE PRECISION     ******  15610C0C D
                                OPR50:                                                                  15620C00 D
0585  1270 0000 0030 0060          Y2FETCH-1=CPCR                %BMAR=Y-ADDRESS                        15630C00 D
0586  24FC 0000 0000 0060          GETF1-1=CPCR                  %F1-FIELD IN A3                        15640000 D
0587  4809 E640 0090 C0F0          A3+AMPCR=AMPCR                                                       15650C00 D
0588  068C 00C0 0030 00C0          OPR50XX-1=AMPCR                                                      15660C00 D
0589  4809 A0C0 8B00 00F0          A1 R=B                        %ISOLATE A-FIELD                       15670C00 D
058A  B070 00C0 0030 00A0          23=SAR;7=LIT                                                         15680C00 D
058B  4824 0F40 9000 C0F0          BMAR=A1;EXEC                                                         15690C00 D
058C  4809 2C56 0C0C 00F0          LIT AND B=MAR                 %READ A(A)                             15700C00 D
058D  A809 00C0 0030 C8F0          MR1;IF RDC                                                           15710C00 D
058E  AC08 0F46 0C3C 00F0          WHEN RDC THEN BMAR+1=MAR,BEX                                         15720C00 D
058F  482D 2F56 000C 00F0          LIT AND BMAR=MAR;JUMP         %A(8)=A(0)                             15730C00 D
                                   %A3=Y-ADDRESS; B=(A(A))                                              15740C00 D
                                %************************     JUMP ON CONDITIONS     ****************** 15750C00 D
                                OPR51:                                                                  15760C00 D
0590  1270 0000 0030 0060          Y2FETCH-1=CPCR                %RETURNS JUMP ADDR. IN BMAR            15770C00 D
0591  24F0 0000 0030 0060          GETF1-1=CPCR                  %RETURNS F1 FOR TYPE 3 IN A3           15780C00 D
0592  4809 E640 0090 C0F0          A3+AMPCR=AMPCR                                                       15790C00 D
0593  0700 0000 0030 00C0          OPR51XX-1=AMPCR               %SETUP FOR SUB OPCODE                  15800C00 D
0594  4809 0F40 1030 C0F0        BMAR=A3                                                                15810C00 D
0595  4809 A000 8B30 00F0        A1 R=B                                                                 15820C00 D
0596  B070 0000 0030 00A0          23 = SAR; 7 = LIT                                                    15830C00 D
0597  4809 2C56 000C 00F0          LIT AND B=MAR                                                        15840C00 D
0598  A824 0000 0030 08F0          EXEC;MR1;IF RDC                                                      15850000 D
0599  AC28 00C0 0C00 C0F0          WHEN RDC THEN BEX;JUMP                                               15860C00 D
                                   %LEAVES (A) IN B; JUMP ADDRESS IN A3                                 15870C00 D
                                % ************************          JUMP AND INDEX        **********     15880C00 D
                                OPR52:                                                                  15890C00 D
059A  1270 0000 0030 0060          Y2FETCH-1=CPCR                %JUMP ADDRESS IN BMAR                  15900C00 D
059B  24FC 0000 000C 0060          GETF1-1=CPCR                  %F1-FIELD IN A3                        15910C00 D
059C  4809 E640 0090 00F0          A3+AMPCR=AMPCR                                                       15920C00 D
059D  0780 0000 0030 00C0          OPR52XX-1=AMPCR                                                      15930C00 D
059E  4809 0F40 0B00 00F0          BMAR=B                        %JUMP ADDRESS IN B                     15940C00 D
059F  4809 A000 9C00 00F0          A1 R=A3                       %ISOLATE B-FIELD                       15950C00 D
05A0  3070 0000 C030 00A0          23=SAR;7=LIT                                                         15960C00 D
05A1  4809 E156 5030 00F0          A3 AND LIT = A3,A1, MIR                                              15970C00 D
05A2  4809 E140 000C 00F0          A3+LIT=MAR                                                           15980C00 D
05A3  0080 0000 0030 00E0          BBASE=LIT                     %MAR1=ADDRESS OF B(A)                  15990C00 D
05A4  A824 0000 0000 08F0          EXEC;MR1;IF RDC                                                      16000C0C D
05A5  AC28 0C40 1C30 00F0          WHEN RDC THEN B=A3,BEX;JUMP                                          16010C00 D
                                   %B=(B(A)); A3=JUMP ADDRESS                                           16020C00 D
                                % ************************       JUMP ON CD          **************     16030C00 D
                                OPR53:                                                                  16040C00 D
05A6  1270 0000 0000 006C          Y2FETCH-1=CPCR                %JUMP ADDR. IN BMAR                    16050C00 D
```

```
05A7  24F0 0000 0030 0060          GETF1-1=CPCR              %RETURNS F1 FIELD IN A3              16060C00 D
05A8  4809 E640 0090 00F0          A3+AMPCR=AMPCR                                                 16070C00 D
05A9  0800 0000 0000 00C0          OPR53XX-1=AMPCR                                                16080C00 D
05AA  4809 A000 9000 00F0          A1 R=A3                                                        16090000 D
05AB  3070 0000 0000 C0A0          23=SAR;7=LIT                                                   16100C00 D
05AC  4824 E156 0B00 C0F0          A3 AND LIT=B;EXEC                                              1611CC00 D
05AD  482D 0F40 1000 00F0          BMAR=A3;JUMP              %JUMP ADDR. IN A3;A-FIELD IN B       16120C00 D
                                % ********************     LOAD CMR TASK              **************1613CC00 D
                                OPR54:                                                            1614C000 D
05AE  1270 00C0 0030 0060          Y2FETCH-1=CPCR            %MIR=(Y)                             1615C000 D
05AF  4809 0C40 0090 00F0       OPR54XX:   B=MIR             %ENTRY FOR OPR55                     16160C00 D
05B0  4809 A0C1 1000 00F0          A1 L=A3                   %ISOLATE AK-FIELDS                   16170C00 D
05B1  2000 0000 0000 0030          COMP 6=SAR                                                     1618CC00 D
05B2  A809 E000 90DC 00FC          A3 R=A3,MAR;IF RDC                                             1619CC00 D
05B3  2400 00C0 0000 00B0          26=SAR;64=LIT             %64= 1C0 OCTAL OFFSET                16200C00 D
05B4  3CD9 E140 109C 00F0          IF LC2 THEN A3+LIT=A3,MAR %LC2=>OPR55 SO OFFEST BY 100         16210C00 D
05B5  2520 0000 0C00 0060          CHECKCMR-1=CPCR           %RETURNS LC1 IF LEGAL                16220C00 D
05B6  2019 0000 0000 00F0          IF NOT LC1 THEN SKIP                                           16230C00 D
05B7  22F0 0000 0000 C040          OUTPUT-1=MPCR                                                  16240C00 D
05B8  221C 0000 000C 0040          ILLEGALCMR-1=MPCR                                              16250C00 D
                                % ********************     STORE CMR TASK             **************1626CC00 D
                                OPR55:                                                            1627CC00 D
05B9  1270 00C0 0000 0060          Y2FETCH-1=CPCR            %B=(Y)                               16280C00 D
05BA  49C9 0000 0000 C0F0          SET LC1                   %PRIVILEGED                          16290C00 D
05BB  4849 0000 003C 00F0          SET LC2                   %SIGNAL TO OPR54 WHOSE CODE IS USED  1630CC00 D
05BC  5AE0 0000 0C30 004C          OPR54XX-1=MPCR                                                 1631CC00 D
                                % ********************     STORE CMR INTERRUPT        **************1632CC00 D
                                OPR56:                                                            16330C00 D
05BD  1270 0000 0000 0060          Y2FETCH-1=CPCR                                                 16340C00 D
05BE  4809 0F40 0090 00F0       OPR56XX:  BMAR=MIR                                                1635CC00 D
05BF  4809 A0C1 1000 00F0          A1 L=A3                   %ISOLATE AK-FIELDS                   16360C00 D
05C0  2000 00C0 0000 C030          COMP 6=SAR                                                     16370C00 D
05C1  A809 E0C0 90DC C0F0          A3 R=MAR,A3; IF RDC                                            16380C00 D
05C2  2400 0000 000C 00B0          26=SAR;64=LIT                                                  16390C00 D
05C3  3CD9 E140 109C C0F0          IF LC2 THEN A3+LIT=A3,MAR %LC2=>OFFSET BY 100 OCTAL            16400000 D
05C4  252C 0000 0C30 006C          CHECKCMR-1=CPCR           %RETURNS LC1 IF LEGAL                16410C00 D
05C5  2E19 0000 0D00 08F0          IF LC1 THEN MR1;BMI;SKIP                                       16420C00 D
05C6  2210 0000 0090 0040          ILLEGALCMR-1=MPCR         %RETURNS TO IFFETCH AFTER PRINTING   16430C00 D
05C7  AC08 0C43 0C1C 00F0          WHEN RDC THEN B=MAR2,BEX                                       16440C00 D
05C8  4809 0C40 0090 00F0          B=MIR                                                          16450C00 D
05C9  237C 00C0 C030 C040          YOUT-1=MPCR                                                    16460C00 D
                                % ********************     LOAD CMR INTERRUPT         **************16470C00 D
                                OPR57:                                                            1648CC00 D
05CA  1270 0000 0030 0060          Y2FETCH-1=CPCR                                                 16490C00 D
05CB  49C9 0000 0030 00F0          SET LC1                   %PRIVILEGED                          16500C00 D
05CC  4849 0000 000C 00F0          SET LC2                   %SIGNALS OPR56 WHOSE CODE IS USED    16510C00 D
05CD  5BDC 0000 0030 C040          OPR56XX-1=MPCR            %SAME CODE OFFESET BY 100 OCTAL      16520C00 D
                                % ********************      STORE CMR IN A REGISTER    ************* 16530000 D
                                OPR60:  %IF I=1THEN OFFSET BY 100 OCTAL                           16540C00 D
05CE  4809 A0C1 1000 20F0          A1 L=A3;ASR               %VOID OPCODE AND SELECT BMAR1        16550C00 D
05CF  2000 0000 0C3C 0030          COMP 6=SAR                                                     16560C00 D
05D0  4809 E000 9030 00F0          A3 R=A3                   %A3=A,F4,B,I                         16570C00 D
05D1  A00C 0000 0030 C020          22=SAR                                                         16580C00 D
05D2  4809 E0C0 80DC 00F0          A3 R=MAR                  %MAR=CMR WITHOUT OFFSET              16590C00 D
05D3  8400 0000 0030 0080          4=SAR;64=LIT              %64=OFFSET OF 100 OCTAL              166C0C00 D
05D4  5FC9 2F40 003C C0F0          IF LST THEN LIT+BMAR=MAR;SET LC1  %IF I=LSB =1 THEN OFFSET     16610C00 D
05D5  2520 0000 0030 006C          CHECKCMR-1=CPCR                                                16620000 D
05D6  A809 E156 9000 08F0          A3 AND LIT R=A3;MR1:IF RDC                                     16630C00 D
05D7  10E0 0000 0000 0080          14=LIT;1=SAR              %MASK B-FIELD AND SAVE               16640C00 D
05D8  AC08 E000 0C3C 00F0          WHEN RDC THEN A3=MAR,BEX  %B=(CMR)                             1665CC00 D
```

```
05D9  4809 0C40 0C30 C0F0          B=MIR                        %SETUP TO WRITE (CMR TO A(B)          16660C00 D
05DA  2819 0000 0C0C 00F0          IF LC1 THEN SKIP                                                   1667CC00 D
05DB  2200 0CC0 003C 0C40          ILLEGALCMR1-1=MPCR                                                 16680C00 D
05DC  0BB0 0000 0000 0040          HALFFETCH-1=MPCR             %MAR=ADDRESS OF A(B)                  16690C00 D
% ********************************      LOAD CMR WITH (A-REG)       **********                       16700C00 D
OPR61:  %IF I=1 THEN OFFSET BY 10C OCTAL                                                             1671CC00 D
05DD  4809 A001 1030 00F0          A1 L=A3                      %VOID OPCODE FIELD                    16720C00 D
05DE  2000 0000 0030 C03C          COMP 6=SAR                                                         16730C00 D
05DF  4809 E000 900C 00F0          A3 R=A3                      %A3=A,F4,B,I                          16740C00 D
05E0  A0EC 0000 C000 00A0          22=SAR;14=LIT                %BMASK IS 14                          16750C00 D
05E1  4809 E156 800C C0F0          A3 AND LIT R=MAR             %MAR=ADDR OF A(B)                     1676CC00 C
05E2  10E0 0000 0000 0080          1=SAR;14=LIT                                                       1677CC00 D
05E3  A809 0000 0C30 C8F0          MR1;IF RDC                                                         16780C00 D
05E4  AC08 E0C0 9C0C 00F0          WHEN RDC THEN A3 R=A3,MAR,BEX                                      16790C00 D
05E5  8400 0000 0C30 0080          4=SAR;64=LIT                                                       16800C00 D
05E6  5FC9 E140 100C 00F0          IF LST THEN A3+LIT=A3,MAR;SET LC1                                  16810000 D
05E7  4809 0C40 0090 00F0          B=MIR                        %B=(AIREG) TO BE WRITTEN              16820C00 D
05E8  252C 0000 0C30 C060          CHECKCMR-1=CPCR              %RETURNS LC1 IF LEGAL                 16830C00 D
05E9  2819 0000 0000 C0F0          IF LC1 THEN SKIP                                                   1684C000 D
05EA  220C 0C00 0030 0040          ILLEGALCMR1-1=MPCR                                                 16850C00 D
05EB  0BB0 0000 0000 0040          HALFFETCH-1=MPCR             %MAR=ADDR OF CMR(AF)                  16860C00 D
% ********************************       SHIFT LEFT CIRCULAR      **************                       16870C00 D
OPR62:  %SHIFT AMOUNT IN SAR;B=(A);MAR= ADDR OF A-REG                                                16880C00 D
05EC  26D0 00C0 0C30 C060          SHIFTAMOUNT-1=CPCR                                                 16890C00 D
05ED  4809 0000 0000 40F0          CSAR                         %COMPLEMENT FOR SHIFT LEFT            16900C00 D
05EE  4809 0C41 8080 C0F0          B C=MIR                                                            16910C00 D
05EF  0BB0 0000 0000 0040          HALFFETCH-1=MPCR             %MAR=A ADDRESS                        16920C00 D
% ********************************    SHIFT LEFT CIRC/DOUBLE       **************16930C00 D
OPR63:                                                                                               16940000 D
        %SAMPLE INPUT A+1=ZZWW, A=XXYY; OUTPUT A+1=WWXX A=YYZZ                                       16950000 D
05F0  26D0 0000 0030 0060          SHIFTAMOUNT-1=CPCR           %B=(A-RE\);MAR=A-ADDR;SAR=SHIFTAMOUNT16960C00 D
05F1  4809 0C40 1000 00F0          B=A3                         %A3=XXYY                              16970C00 D
05F2  4809 0019 0B00 40F0          B111 L=B,CSAR                %B=FF00                               16980C00 D
05F3  4809 0F46 0C0C 00F0          BMAR+1=MAR                   %SETUP FOR A+1                        16990C00 D
05F4  4809 2F56 000C C0F0          LIT AND BMAR=MAR             %CONVERT A(8) TO A(0)                 17000C00 D
05F5  0070 00C0 0030 00E0          7=LIT                                                              17010C00 D
05F6  A809 EC56 8080 08F0          A3 AND B R=MIR;MR1;IF RDC    %MIR=00XX                             17020C00 D
05F7  AC08 E001 1C0C C0F0          WHEN RDC THEN A3 L=A3,BEX    %B=ZZWW;A3=YY0C                       17030C00 D
05F8  4809 0C41 0F3C C0F0          B L=BBI                      %B=WWXX                               17040C00 D
05F9  4809 0C40 0C9C 00F0          B=MIR,BEX                    %B=ZZWW;7IR=WWXX=NEW(A+1)             17050C00 D
05FA  2019 0000 0C3C C0F0          IF NOT LC1 THEN SKIP                                               17060C00 D
05FB  0F20 00C0 0000 C040          OPR63XX-1=MPCR                                                     17070C00 D
05FC  9809 0C40 8BB0 18F0          B R=B;MW1;IF SAI             %B=00ZZ                               17080C0C D
05FD  9C08 0B02 803C 00F0          WHEN SAI THEN NOT CTR R=MAR                                        17090C00 D
05FE  0000 00C0 0C30 003C          24=SAR                                                             17100C00 D
05FF  4809 EC5C 0090 00F0          A3 OR B=MIR                  %MIR=YYZZ                             17110C00 D
0600  0BB0 0000 0C30 0040          HALFFETCH-1=MPCR             %(A)=YYZZ                             17120C00 D
0601  4809 0C40 8B30 00F0  OPR63XX:   B R=B                                                          17130C00 D
0602  4809 EC5C 0D90 00F0          A3 OR B=MIR,BMI                                                    17140C00 D
0603  9809 0000 0000 18F0          MW1;IF SAI                                                         17150C00 D
0604  9C08 0C40 0C90 00F0          WHEN SAI THEN B=MIR                                                17160C00 D
0605  4809 0BC2 803C 00F0          NOT CTR R=MAR                                                      17170C00 D
0606  0000 0000 0C30 0030          24=SAR                                                             17180C00 D
0607  0BBC 0000 0000 0040          HALFFETCH-1=MPCR                                                   1719CC00 D
% ********************************       SHIFT RIGHT FILL WITH ZEROS     ********                  17200C00 D
OPR64:      %SAR=SHIFT AMOUNT;B=(A);MAR=A ADDRESS                                                    17210C00 D
0608  26D0 00C0 0030 C060          SHIFTAMOUNT-1=CPCR                                                 17220C00 D
0609  4809 0C40 8090 00F0          B R=MIR                                                            17230000 D
060A  0BB0 00C0 0C30 0040          HALFFETCH-1=MPCR             %MAR=A-REG ADDRESS                    17240C00 D
% ********************************     SHIFT RIGHT DBL/FILL ZEROS    **************17250C00 D
```

```
                                                OPR65:
                                                                                                  17260C00 D
                                                      %EXAMPLE INPUT A+1=ZZWW, A=XXYY; OUPUT A+1=00ZZ, A=WWXX      17270C00 0
C60B  2600 00C0 0090 0060                             SHIFTAMOUNT-1=CPCR          %B=A-REG;MAR=A-ADDR;SAR=SHIFT AMOUNT 17280CCC D
060C  4809 0F46 0C9C 00FC                             BMAR+1=MAR                  %READ A+1                        17290C0C 0
C60D  4809 2F56 000C 00F0                             LIT AND BMAR=MAR            %CONVERT A(8) TO A(0)            17300C00 D
060E  0070 0CC0 0090 C0E0                             7=LIT                                                        17310C90 0
060F  2019 00C0 0090 00FC                             IF NOT LC1 THEN SKIP                                         17320C00 0
0610  0F3C 0C00 0000 0040                             OPR65XX-1=MPCR                                               17330C00 D
0611  A809 0C40 900C 08F0                             B R=A3;MR1;IF ROC           %A3=00XX                         17340C00 0
0612  AC08 0000 0C00 00F0                             WHEN RDC THEN BEX           %B=ZZWW                          17350C00 0
0613  4809 0C40 8C90 00F0                             B R=MIR,BEX                 %MIR=00ZZ;A3=00XX;B=ZZWW         17360C00 D
0614  9809 CC41 0B30 18F0                             B L=B;MW1;IF SAI                                             17370000 D
0615  9E08 EC5C 0090 20F0                             WHEN SAI THEN A3 OR 0=MIR;ASR                                17380000 0
0616  4809 CBC2 8C9C 00F0                             NOT CTR R=MAR               %RESET A-ADDRESS                 17390C00 0
0617  0000 0000 0090 0030                             24=SAR                      %B=WWXX                          17400C00 D
0618  0BBC 0000 0C00 C040                             HALFFETCH-1=MPCR                                             17410C00 0
0619  A809 0000 CC90 08F0                   OPR65XX:  0=MIR;MR1;IF ROC                                             17420C0C 0
061A  AC08 0000 0C30 C0F0                             WHEN RDC THEN BEX                                            17430C00 D
061B  9809 0000 0090 18F0                             MW1;IF SAI                                                   17440000 0
061C  9C08 0C40 8090 00F0                             WHEN SAI THEN B R=MIR                                        17450C90 D
061D  4809 CBC2 809C C0F0                             NOT CTR R=MAR                                                17460C00 0
C61E  0000 00C0 0090 0030                             24=SAR                                                       17470C00 D
061F  0BB0 00C0 0000 C040                             HALFFETCH-1=MPCR                                             17480C00 0
                                                % ********************      SHIFT RIGHT SIGN FILL      ************** 17490C00 D
                                                OPR66:      %SAR=SHIFT AMOUNT;MAR=AIREG ADDR; B=(A)                  17500C00 0
0620  2600 0000 CC0C 0060                             SHIFTAMOUNT-1=CPCR                                           17510C0C 0
C621  4809 0C40 8080 40F0                             B R=MIR,CSAR                %SAR SET FOR SIGN 1 EXTENSION    17520C00 0
0622  0BB0 0000 0090 00CC                             HALFFETCH-1=AMPCR                                            17530C00 0
0623  4C00 0019 0F00 00F0                             IF MST THEN B111 L=BBI;STEP ELSE JUMP                        17540000 D
0624  4820 0C40 0090 C0F0                             B=MIR;JUMP                  %NEGATIVE SIGN EXTENSION         17550C00 0
                                                % ********************    SHIFT RIGHT DBLE/FILL SIGN   ************17560C00 0
                                                OPR67:                                                           17570C00 0
                                                      %EXAMPLE INPUT A+1=ZZWW,A=XXYY; OUT&UT=A+1=00ZZ,A=WWXX      17580C00 D
                                                      %EXAMPLE INPUT=-ZZWW;A=XXYY; OUTPUT=A+1=FFZZ, A=WWXX       17590C0C 0
0625  2600 00C0 0090 0060                             SHIFTAMOUNT-1=CPCR                                           17600C00 D
C626  4809 0F46 009C 00F0                             BMAR+1=MAR                                                   17610C00 0
C627  4809 2F56 000C 00F0                             LIT AND BMAR=MAR            %CONVERT A(8) TO A(0)            17620C00 0
0628  007C 0000 0000 00E0                             7=LIT                                                        17630C00 D
C629  2019 0000 0000 00F0                             IF NOT LC1 THEN SKIP                                         17640C00 0
C62A  0F40 0000 000C 0040                             OPR67XX-1=MPCR                                               17650C00 0
062B  A809 0C40 9C00 08F0                             B R=A3;MR1;IF ROC           %A3=00XX                         17660C00 D
062C  AC08 0000 0C0C 00F0                             WHEN ROC THEN BEX           %B=ZZWW                          17670C00 D
062D  4809 0C40 8090 C0F0                             B R=MIR                     %MIR=00ZZ                        17680C00 0
062E  4C0B 0C19 0F00 C0FC                             IF MST THEN B111 L=BBI; STEP ELSE SKIP  %CHECK FOR NEGATIVE 17690C00 0
062F  4809 CC40 0C90 C0F0                             B=MIR,BEX                   %B=ZZWW;A3=00XX;MIR=0CZZ         17700C00 0
0630  9809 0C41 0B30 18F0                             B L=B;MW1;IF SAI            %B=WW0C                          17710C0C D
0631  9E08 EC5C 0090 20F0                             WHEN SAI THEN A3 OR B=MIR;ASR  %MIR=WWXX                     17720C00 0
C632  4809 0B02 809C 00F0                             NOT CTR R=MAR               %RESET A-ADDRESS                 17730C00 0
0633  0000 00C0 0090 0030                             24=SAR                      %OUTPUT=WWXX                     17740C00 0
0634  0BB0 0000 0090 C040                             HALFFETCH-1=MPCR            %A+1=SIGC/ZZ                     17750C00 0
                                                OPR67XX:   %B=(A);SAR=SHIFT-32                                   17760C00 0
0635  A809 0000 0000 C8F0                             MR1;IF ROC                                                   17770C00 D
C636  ACC8 0000 1C00 00F0                             WHEN RDC THEN 0=A3,BEX      %B=(A+1);MIR=(A SHIFTED)         17780C00 D
0637  4809 0C40 8C90 00F0                             B R=MIR                                                      17790C00 C
0638  4C0B 0DC2 1000 00F0                             IF MST THEN NOT 0=A3;STEP ELSE SKIP                          17800C00 D
0639  481B 0019 0F00 00F0                             B111 L=BBI;SKIP                                              17810C00 D
063A  4809 0000 0090 C0F0                             BMI                                                          17820C00 0
063B  4809 E0C0 0090 C0FC                             A3=MIR                                                       17830C00 0
063C  9809 0000 000C 18F0                             MW1;IF SAI                                                   17840C00 D
C630  9C08 0B02 8C9C 00F0                             WHEN SAI THEN NOT CTR R=MAR                                  17850C00 0
```

```
063E  0000 0000 0090 0030          24=SAR                                                    17860C00 D
063F  4809 0C40 0090 00FC          B=MIR                                                     17870C00 D
0640  0BB0 0000 0090 0040          HALFFETCH-1=MPCR                                          17880C00 D
                                % ************************    FLOATING POINT SETUPS          ***********  17890000 D
                                OPR70:      %RETURN WITH B=(A);BMAR=A-REG ADDRESS           17900C00 D
0641  24C0 0000 0090 0060          GETAA3-1=CPCR               %MAR; A3=A-FIELD              17910C00 D
0642  2460 00C0 0090 0060          GETF-1=CPCR                 %RETURNS F2-FIELD IN A3       17920C00 D
0643  4809 E640 0040 00F0          A3 + AMPCR=AMPCR                                          17930C00 D
0644  0870 0000 0090 00C0          OPR70XX-1=AMPCR                                           17940C00 D
0645  A809 0000 0000 08F0          MR1;IF ROC                                                17950000 D
0646  4824 0003 0090 00F0          EXEC                                                      17960C00 D
0647  AC28 0000 0C00 00F0          WHEN ROC THEN BEX;JUMP      %B=(A);MAR=A-REG ADDRESS      17970C00 D
                                % ************************       HALFWORD LOGICALS AND ARITH      ******  17980C00 D
                                OPR71:      %B=(A(B));A3=(A(A));MAR=A-FIELD ADDR             1799CC00 D
0648  2460 00C0 0C00 0060          GETF-1=CPCR                 %RETURNS F2-FIELD IN A3       18000C00 D
0649  4809 E640 0040 00F0          A3+AMPCR=AMPCR                                            18010C00 D
064A  0F50 00C0 000C 00C0          OPR71XX-1=AMPCR                                           18020C00 D
064B  4809 A000 9030 C0F0          A1 R=A3                                                   18030C00 D
064C  1070 0003 0000 00A0          17=SAR;7=LIT                                              18040C00 D
064D  4809 E156 000C 00F0          A3 AND LIT=MAR              %ISOLATE B FOR A(B)           18050C00 D
064E  A809 A0C0 9000 08F0          A1 R=A3;MR1;IF ROC          %READ A(B)                    18060C00 D
064F  B000 0000 0090 0020          23=SAR                      %ISOLATE A-FIELD              18070C00 D
0650  AC08 E156 0C0C 00F0          WHEN ROC THEN A3 AND LIT=MAR,BEX                          18080C00 D
0651  A809 0C40 0090 C8F0          B=MIR;MR1;IF ROC            %MIR=A(B)                     18090C00 D
0652  AC08 0000 0C90 00F0          WHEN ROC THEN BEX                                         18100C00 D
0653  2809 0C40 1000 00F0          B=A3,BMI;IF LC1             %A3=A(A);B=A(B);CLEAR LC1     18110C00 D
0654  3824 0000 0090 00F0          EXEC;IF LC2                 %EXECUTE OPR71XX + F-FIELD;CLEAR LC2 18120000 D
0655  4809 0000 C090 00F0          0;STEP                      %RESET AOV AND ABT FLAGS      18130C00 D
0656  2C0B EC40 0090 00F0          IF LC1 THEN A3+B=MIR;STEP ELSE SKIP %OPR71X1              18140C00 D
0657  7CC9 EC46 0090 00F0          IF AOV THEN A3+B+1=MIR                                    18150C00 D
0658  5C09 00C0 0090 00F0          IF ABT THEN 0=MIR           %ELIMINATE NEGATIVE 0         18160C00 D
0659  3C0B EC5E 0C90 C0FC          IF LC2 THEN A3-B=MIR;STEP ELSE SKIP %OPR71X2              18170C00 D
065A  7409 EC58 0090 00F0          IF NOT AOV THEN A3-B-1=MIR                                18180C00 D
065B  0BB0 0003 0090 0040          HALFFETCH-1=MPCR            %MAR=A-REG ADDRESS            18190C00 D
                                % ************************     HALFWORD LOGICAL SUBCODES      **********  18200000 D
                                OPR71XX:                                                    18210C00 D
065C  4809 EC5C 0090 00F0          A3 OR B=MIR                 %OPR71X0                      18220C00 D
065D  39C9 0C00 0090 00F0          SET LC1;IF LC2                                            18230C00 D
065E  2849 00C0 0000 00F0          SET LC2;IF LC1                                            18240C00 D
065F  4809 EC4C 0090 C0F0          A3 XOR B=MIR                %OPR71X3                      18250C00 D
0660  21DC 0000 0000 C040          ILLEGAL-1=MPCR              %INVALID INSTRUCTION          18260C00 D
0661  4809 EC56 0C90 00F0          A3 AND B=MIR                %OPR71X5                      18270000 D
                                % ************************     REGISTER OPERATIONS HALFWORD     ********  18280C00 D
                                OPR74:  %PROVIDES B,MIR=A-FIELD; A3=B-FIELD                 18290C0C D
0662  2460 0000 0000 0060          GETF-1=CPCR                 %RETURNS F2-FIELD IN A3       183C0C0C D
0663  4809 E640 0040 00F0          A3+AMPCR=AMPCR              %SETUP SUBOPCODE              18310C00 D
0664  08FC 0000 0090 00C0          OPR74XX-1=AMPCR                                           18320C00 D
0665  4809 A0C0 9000 0CF0          A1 R=A3                                                   18330C00 D
0666  B070 0000 0C00 00A0          23=SAR;7=LIT                                              18340C00 D
0667  4824 E156 0B90 00F0          A3 AND LIT=MIR,B;EXEC                                     18350C00 D
0668  4809 A000 9C00 00F0          A1 R=A3                                                   18360C00 D
0669  1000 00C0 0090 C020          17=SAR                      %ISOLATE B-FIELD              18370C00 D
066A  482D E156 1090 00F0          A3 AND LIT=A3;JUMP                                        18380C00 D
                                % ************************     INTERRUPT HANDLERS           **********  18390000 D
                                OPR77:                                                      18400C00 D
066B  2460 0000 0090 0060          GETF-1=CPCR                 %A3=F-FIELD                   18410C00 D
066C  4809 E640 0040 00F0          A3+AMPCR=AMPCR                                            18420C00 D
066D  097C 0000 0090 00C0          OPR77XX-1=AMPCR                                           18430C00 D
066E  4809 0000 0090 00FC          STEP                        %ALLOW TIME FOR AMPCR TO SET  18440C00 D
066F  4824 00C0 0C00 00F0          EXEC                                                      18450C00 D
```

```
0670  482D 0000 0000 00F0          JUMP                        %JUMP TO SUBOPCODE IN AMPCR          18460000 D
                                % ********************************      INCLUSIVE OR  (A OR )       **********          18470000 D
                                OPR01X0:                          %INCLUSIVE Y OR A => A                             18480000 D
0671  4809 EC5C 0090 00F0          A3 OR B = MIR                                                                        18490000 D
0672  2410 0000 0000 0040          OUTLOGIC-1=MPCR                                                                      18500000 D
                                % ********************************      SELECTIVE CLEAR A WITH Y=>A(A+1)  ******          18510000 D
                                OPR01X1:                                                                             18520000 D
                                   %SELECTIVE CLEAR A, A AND NOT Y = A                                               18530000 D
0673  4809 EC44 0090 00F0          A3 NRI B = MIR                                                                       18540000 D
0674  2410 0000 0000 0040          OUTLOGIC-1=MPCR                                                                      18550000 D
                                % ********************************        SELECTIVE SUBSTITUTE   **************          18560000 D
                                OPR01X2:    %MAR=ADDR OF A-REG; B=A(A);A3=(Y)                                        18570000 D
0675  4809 0F46 000C 00F0          BMAR+1=MAR                                                                           18580000 D
0676  4809 2F56 000C 00F0          LIT AND BMAR=MAR              %CONVERT A(8) TO A(0)                                  18590000 D
0677  A809 E000 0090 08F0          A3=MIR;MR1;IF RDC             %READA(A+1)                                            18600000 D
0678  AC08 0C42 1C00 00F0          WHEN RDC THEN NOT B=A3,BEX                                                           18610000 D
                                   %A3=NOT (A); B=(A+1); MIR=(Y)                                                     18620000 D
0679  4809 EC56 0D90 00F0          A3 AND B=MIR,BMI                                                                     18630000 D
                                   %B=(Y); MIR=NOT (A) AND (A+1); A3=NOT (A)                                         18640000 D
067A  4809 EC49 0F00 00F0          A3 NRI B=BBI                  %A3=(A) AND B= (Y)  OR MIR=NOT (A)                     18650000 D
067B  4809 0C40 0090 00F0          B=MIR                                                                                18660000 D
067C  2410 0000 0000 0040          OUTLOGIC-1=MPCR               % AND (A+1)                                            18670000 D
                                % ********************************     EXCLUSIVE OR   A XOR Y         **********          18680000 D
                                OPR01X3:                                                                             18690000 D
                                   % A XOR Y = A                                                                     18700000 D
067D  4809 EC4C 0090 00F0          A3 XOR B = MIR                                                                       18710000 D
067E  2410 0000 0000 0040          OUTLOGIC-1=MPCR                                                                      18720000 D
                                % ********************************     AND  A(A+1)+A(A) AND Y => A(A+1)    ******          18730000 D
                                OPR01X4:                                                                             18740000 C
                                   %A+1 + A AND Y =>A+1                                                              18750000 D
067F  4809 0F46 000C 00F0          BMAR+1=MAR                                                                           18760000 D
0680  4809 2F56 000C 00F0          LIT AND BMAR=MAR              %CONVERT A(8) TO A(0)                                  18770000 D
0681  A809 EC56 1000 08F0          A3 AND B=A3;MR1;IF RDC                                                               18780000 D
0682  AC08 00C0 0C0C 00F0          WHEN RDC THEN BEX             %A+1=>B                                                18790000 D
0683  4809 EC40 0C90 00F0          A3+B=MIR                      %STORE ANSWER IN A+1                                   18800000 D
0684  7C09 EC46 0090 00F0          IF AOV THEN A3+B+1=MIR                                                               18810000 D
0685  6C09 0000 0090 00F0          IF ABT THEN 0=MIR             %ELIMINATE NEGATIVE 0                                  18820000 D
0686  2410 0000 000C 0040          OUTLOGIC-1=MPCR                                                                      18830000 D
                                % ********************************     AND  A(A) AND Y => A(A)        **********          18840000 D
                                OPR01X5:                                                                             18850000 D
                                   %A AND Y =>A                                                                      18860000 D
0687  4809 EC56 0090 00F0          A3 AND B=MIR                                                                         18870000 D
0688  22F0 0000 0090 0040          OUTPUT-1=MPCR                                                                        18880000 D
                                % ********************************     AND SUBTRACT A(A+1)-Y AND A(A) => A(A+1) *          18890000 D
                                OPR01X6:                                                                             18900000 D
                                   %A+1 - A AND Y => A+1                                                             18910000 D
0689  4809 0F46 000C 00F0          BMAR+1=MAR                                                                           18920000 D
068A  4809 2F56 000C 00F0          LIT AND BMAR=MAR              %CONVERT A(8) TO A(0)                                  18930000 D
068B  A809 EC56 0090 08F0          A3 AND B = MIR;MR1;IF RDC                                                            18940000 D
068C  AC08 0000 0C00 00F0          WHEN RDC THEN BEX                                                                    18950000 D
068D  4809 0C40 1D00 00F0          B=A3,BMI                      %SWAP REGISTERS TO SUBTRACT                            18960000 D
068E  4809 EC5E 0090 00F0          A3-B=MIR                                                                             18970000 D
068F  7409 EC58 0090 00F0          IF NOT AOV THEN A3-B-1=MIR                                                           18980000 D
0690  2410 0000 0000 0040          OUTLOGIC-1=MPCR                                                                      18990000 D
                                % ********************************     AND  A(A) AND Y => A(A+1)      **********          19000000 D
                                OPR01X7:                                                                             19010000 D
                                   %A AND Y => A+1                                                                   19020000 D
0691  4809 0F46 000C 00F0          BMAR+1=MAR                                                                           19030000 D
0692  4809 2F56 000C 00F0          LIT AND BMAR=MAR              %CONVERT A(8) TO A(0)                                  19040000 D
0693  4809 EC56 0090 00F0          A3 AND B=MIR                                                                         19050000 D
```

```
0694  2410 0000 0000 0040          OUTLOGIC-1=MPCR                                                    19060C00 D
                                % ************************    COUNT ONES                    ************19070C0C D
                                OPR02X0:                            % Y=>A3, A(A)=>B                    19080C00 D
0695  4809 E0C0 0B30 00F0          A3 = B                                                             19090C00 D
0696  2920 0000 0000 C060          COUNTONES-1=CPCR                                                   19100C00 D
0697  22FC 0000 0030 0040          OUTPUT-1 = MPCR                                                    19110000 D
                                % ************************    EXECUTE REMOTE       ****************** 19120000 D
                                OPR02X2:                            % Y VALUE => A3                     19130C00 D
0698  4809 E000 0B30 00F0          A3=B                           %REMOTE INSTR. SETUP FOR IFETCH     19140C00 D
0699  0ADC 0CC0 0000 0040          IFETCHREMOTE-1=MPCR            %LET IFETCH EXECUTE                 19150C00 D
                                % ************************    EXECUTE REMOTE LOWER ****************** 19160C00 D
                                OPR02X3:                            % A3 = Y VALUE                      19170C00 D
069A  4809 E0C0 8C40 C0F0          A3 R=AMPCR                     %SETUP OPCODE                       19180C00 D
069B  23F0 0003 0030 C090          63=LIT;10=SAR                                                      19190C00 D
069C  4809 2656 0040 C0F0          LIT AND AMPCR=AMPCR                                                19200C00 D
069D  4809 E001 4030 00F0          A3 L=A1                                                            19210C00 D
069E  0000 0000 0000 C020          16 = SAR                                                           19220C0C D
069F  4824 C001 8B00 40F0          A2 C=B,CSAR;EXEC               %SET LOWER HALF BIT ON              19230C00 D
06A0  8000 0000 0C30 0020          20=SAR                                                             19240CC0 D
06A1  482D 0CC1 A030 00F0          BIT1 C=A2;JUMP                 %RESET ASR TO NORMAL                19250C00 D
                                % ************************    A(A+1) AND Y => Y          ************** 19260C00 D
                                OPR02X4:                            % A(A)=>B, Y ADDR => MIR            19270C00 D
06A2  4809 0F46 0C0C 00F0          BMAR+1 = MAR                   % A(A+1)                             19280C00 D
06A3  4809 2F56 000C C0FC          LIT AND BMAR=MAR               %CONVERT A(8) TO A(0)               19290C00 D
06A4  A809 0000 0000 08F0          MR1; IF RDC                                                        19300C00 D
06A5  AC08 0C40 1C00 00F0          WHEN RDC THEN B=A3,BEX         % A(A)=>A3, A(A+1)=>B                19310C00 D
06A6  4809 EC56 0D30 00F0          A3 AND B = MIR,BMI                                                 19320C00 D
06A7  4809 0C43 0C1C 00F0          B = MAR2                                                           19330C00 D
06A8  237C 0000 0030 0040          YOUT-1=MPCR                                                        19340C00 D
                                % ************************    A(A)+A(A+1) => A(A+1) & Y     ********** 19350C00 D
                                OPR02X5:                            % A(A)=>B, Y ADDR => MIR            19360C00 D
06A9  4809 0F46 000C 00F0          BMAR+1 = MAR                   % A(A+1)                             19370C00 D
06AA  4809 2F56 000C 00F0          LIT AND BMAR=MAR               %CONVERT A(8) TO A(0)               19380C00 D
06AB  A809 0000 000C C8F0          MR1; IF RDC                                                        19390C00 D
06AC  AC08 0C40 1C00 00F0          WHEN RDC THEN B = A3,BEX                                           19400C00 D
06AD  4809 EC40 0B30 00F0          A3+B=B                                                             19410C00 D
06AE  7C09 0C46 0B00 00F0          IF AOV THEN B+1=B                                                  19420C00 D
06AF  6C09 0C42 0B00 00F0          IF ABT THEN NOT B=B                                                19430C00 D
06B0  4809 0C40 0D30 00F0          B=MIR,BMI                                                          19440C00 D
06B1  2300 0000 0000 006C          OUTPUT1-1 = CPCR               % A(A)+A(A+1) => A(A+1)              19450C00 D
06B2  4809 0C43 001C 00F0          B = MAR2                                                           19460C00 D
06B3  2370 0000 0030 0040          YOUT-1=MPCR                                                        19470C00 D
                                % ************************    A(A+1)-A(A)   => A(A+1) & Y  ********** 19480C00 D
                                OPR02X6:                            % A(A)=>B,ADDR Y => MIR             19490C00 D
06B4  4809 0F46 000C C0F0          BMAR+1 = MAR                   % A(A+1)                             19500000 D
06B5  4809 2F56 000C 00F0          LIT AND BMAR=MAR               %CONVERT A(8) TO A(0)               19510C00 D
06B6  A809 0000 0C00 C8F0          MR1; IF RDC                                                        19520C00 D
06B7  AC08 0C40 1C00 C0F0          WHEN RDC THEN B=A3,DEX         %   A(A)=>A3, A(A+1)=> B            19530C00 D
06B8  4809 CC40 4030 00F0          B = A1                                                             19540C00 D
06B9  4809 E000 0B3C C0F0          A3 = B                                                             19550C00 D
06BA  4809 AC5E 4C30 00F0          A1-B=A1                                                            19560C00 D
06BB  7409 A0DE 4030 C0F0          IF NOT AOV THEN A1-1=A1                                            19570C00 D
06BC  4809 A000 0D30 C0F0          A1=MIR,BMI                                                         19580C00 D
06BD  2300 0000 0030 C060          OUTPUT1-1=CPCR                                                     19590C00 D
06BE  4809 0C43 001C 00F0          B = MAR2                                                           19600C00 D
06BF  237C 0000 0030 0040          YOUT-1=MPCR                                                        19610C00 D
                                % ************************    DOUBLE STORE A             ********** 19620C00 D
                                OPR02X7:  %B=(A(A));A3=(Y);MIR=Y-ADDRESS                           19630C00 D
06C0  4809 0F46 000C 00F0          BMAR+1=MAR                     %MAR=A+1                            19640C0C D
06C1  4809 2F56 000C 00F0          LIT AND BMAR=MAR               %CONVERT A(8) TO A(0)               19650C00 D
```

```
06C2  A809 0C40 0D30 C8F0          B=MIR,BMI;MR1;IF RDC        %MIR=A(A);B=Y-ADDR                    19660C00 D
06C3  AC08 0C40 0C1C 00F0          WHEN RDC THEN B=MAR2,BEX    %B=(A(A+1));MAR2=Y                    19670C00 D
06C4  9809 00C0 0030 1CF0          MW2;IF SAI                                                        19680C00 D
06C5  9C08 0F46 001C 00F0          WHEN SAI THEN BMAR+1=MAR2                                         19690C00 D
06C6  4809 0C40 0C30 00F0          B=MIR                       %SETUP A+1 FOR OUTPUT                 19700C00 D
06C7  2320 0000 0030 0040          OUTPUT2NORET-1=MPCR                                               19710C00 C
                                 % ********************      TEST ANC SET FLAG      ***************** 19720C00 D
                                 OPR03X7:                                                            19730C00 C
06C8  127C 0000 000C 0060          Y2FETCH-1=CPCR              %BMAR=YADDRESS; B=(Y)                 19740C00 D
06C9  4809 0C40 0030 C0F0          B                           %TEST MOST SIGNIFICANT BIT            19750C00 D
06CA  4419 1C40 0030 00F0          IF NOT MST THEN B1TT=MIR;SKIP                                     19760C00 D
06CB  0F6C 0000 0030 0040          OPR03XX7-1=MPCR                                                   19770C00 D
06CC  9809 00C0 0000 1CF0          MW2;IF SAI                                                        19780C00 C
06CD  9C18 D81C 2C00 00F0          WHEN SAI THEN A2 OR B100=A2;SKIP                                  19790C00 D
06CE  4809 D80E 2C30 C0F0  OPR03XX7:  A2 AND B011=A2                                                 19800C00 D
06CF  0A50 0000 0030 0040          IFETCH-1=MPCR                                                     19810C00 D
                                 % ********************      DOUBLE LOAD A                 ********** 19820C00 C
                                 OPR05X0:   %MIR=(Y);MAR2=YADDR                                      19830C00 C
06D0  4809 0F46 001C 00F0          BMAR+1=MAR2                 %READ Y+1                             19840C00 D
06D1  A809 0000 0000 0CF0          MR2;IF RDC                                                        19850C00 D
06D2  AC08 E0C0 0C3C 00F0          WHEN RDC THEN A3=MAR,BEX    %A3=A-ADDRESS                         19860C00 D
06D3  A809 00C0 0030 18F0          MW1;IF RDC                  %WRITE Y                              19870C00 D
06D4  9C08 E0C0 003C C0F0          WHEN SAI THEN A3+1=MAR                                            19880C00 D
06D5  4809 2F56 000C 00F0          LIT AND BMAR=MAR            %CONVERT A(8) TO A(0)                 19890C00 D
06D6  4809 0C40 0C30 00F0          B=MIR                                                             19900C00 D
06D7  22F0 0000 0000 0040          OUTPUT-1=MPCR               %WRITE Y+1                            19910C00 D
                                 % ********************      DOUBLE ADD                 ********** 19920C00 D
                                 OPR05X1:  %A3=A-ADDRESS;MAR2=Y-ADDRESS;MIR=(Y)                      19930C00 D
06D8  4809 0F46 0D1C C0F0          BMAR+1=MAR2,BMI                                                   19940C00 D
06D9  A809 0C40 4000 CCF0          B=A1;MR2;IF RDC             %A1=(Y)                               19950C00 D
06DA  AC08 E0C0 0C3C 00F0          WHEN RDC THEN A3=MAR,BEX                                          19960C00 D
06DB  A809 0C40 1000 08F0          B=A3;MR1;IF RDC             %A3=(Y+1)                             19970C00 D
06DC  AC08 0000 0C30 00F0          WHEN RDC THEN BEX           %B=(A)                                19980C00 D
06DD  4809 AC40 4030 00FC          A1+B=A1;MIR                 %A1=A+Y                               19990C00 D
06DE  7BC9 0F40 0B00 00F0          BMAR=B; IF AOV THEN SET LC1 %TEST FOR CARRY                       20000C00 D
06DF  4809 0F46 000C 00F0          BMAR+1=MAR                                                        20010C00 D
06E0  4809 2F56 000C C0F0          LIT AND BMAR = MAR          %CONVERT A(8) TO A(0)                 20020000 D
06E1  A809 0000 0030 08F0          MR1;IF RDC                                                        20030C00 D
06E2  AC08 0C40 0C3C 00F0          WHEN RDC THEN B=MAR,BEX                                           20040C00 D
06E3  2C19 EC46 100C 00F0          IF LC1 THEN A3+B+1=A3;SKIP %ADD CARRY                             20050C00 D
06E4  4809 EC40 103C C0F0          A3+B=A3                                                           20060000 D
06E5  7C09 ACC0 4030 C0F0          IF AOV THEN A1+1=A1,MIR     %ADD CARRY                            20070C00 D
06E6  4809 E0C2 0000 00F0          NOT A3                                                            20080C00 D
06E7  6C0B A0C2 0000 00F0          IF ABT THEN NOT A1;STEP ELSE SKIP                                 20090C00 D
06E8  6C09 0000 5030 C0F0          IF ABT THEN 0=A1,A3,MIR                                           20100C00 D
06E9  9809 0000 0030 18F0          MW1;IF SAI                  %OUTPUT A+Y                           20110C00 D
06EA  9C08 0F46 000C 00F0          WHEN SAI THEN BMAR+1=MAR                                          20120C00 D
06EB  4809 2F56 003C 00F0          LIT AND BMAR=MAR            %CONVERT A(8) TO A(0)                 20130C00 D
06EC  4809 E000 0030 00F0          A3=MIR                                                            20140C00 D
06ED  22F0 00C0 0000 0040          OUTPUT-1=MPCR               %OUTPUT (A+1)+(Y+1)                   20150C00 D
                                 % ********************      DOUBLE SUBTRACT (A+1,A)-(Y+1,Y)=>A+1,A ***20160C00 D
                                 OPR05X2:  %A3=A-ADDR;MAR2=Y-ADDR;MIR=(Y)                            20170C00 C
06EE  4809 0F46 0D1C 00F0          BMAR+1=MAR2,BMI             %B=(Y)                                20180C00 D
06EF  A809 0C40 4000 0CF0          B=A1;MR2;IF RDC             %A1=(Y)                               20190C00 C
06F0  AC08 E0C0 0C3C C0F0          WHEN RDC THEN A3+1=MAR,BEX                                        20200C00 D
06F1  4809 2F56 000C 00F0          LIT AND BMAR=MAR            %CONVERT A(8) TO A(0)                 20210C00 D
06F2  A809 0C40 0030 08F0          B=MIR;MR1;IF RDC            %MIR=(Y+1);READ A+1                   20220000 D
06F3  AC08 E0C0 0C0C 00F0          WHEN RDC THEN A3=MAR,BEX    %B=(A+1)                              20230C00 D
06F4  4809 0C40 1D30 00F0          B=A3,BMI                    %A3=(A+1)                             20240C00 D
06F5. A809 EC5E 1030 08F0          A3-B=A3;MR1;IF RDC          %A3=(A+1)-(Y+1)                       20250C00 D
```

```
06F6  73C9 0000 0090 C0FC          IF NOT AOV THEN SET LC1                                          2C26CC00 D
06F7  AC08 A0C0 0C90 C0F0          WHEN RDC THEN A1=MIR,BEX   %B=(A)                                2027CC00 D
06F8  4809 0C40 4090 C0F0          B=A1,BMI                                                         2028CC00 D
06F9  2C19 AC58 4090 C0F0          IF LC1 THEN A1-B-1=A1,MIR;SKIP                                   2029CC00 D
06FA  4809 AC5E 4090 00F0          A1-B=A1,MIR                                                      2030CC00 D
06FB  740B E0DE 1000 00F0          IF NOT AOV THEN A3-1=A3;STEP ELSE SKIP                           2031CC00 D
06FC  7409 A0DE 4090 00FC          IF NOT AOV THEN A1-1=A1,MIR                                      2032CC0C D
06FD  9809 00C0 0C00 18F0          MW1;IF SAI                                                       2033CC00 D
06FE  9C08 0F46 000C 00F0          WHEN SAI THEN BMAR+1=MAR   %WRITE A                              2034CC00 D
06FF  4809 2F56 000C 00F0          LIT AND BMAR=MAR           %CONVERT A(8) TO A(0)                 2035CC00 D
0700  4809 E0C0 0090 00F0          A3=MIR                     %WRITE A+1                            2036CC0C D
0701  22FC 0000 0C0C 0040          OUTPUT-1=MPCR                                                    2037CC00 D
                                 % ********************************     DOUBLE COMPARE                ************2038CC00 D
                                 OPR05X3:                                                           2039CC00 D
0702  4809 0F46 001C C0F0          BMAR+1=MAR2                %MAR2=Y-ADDR;MIR=(Y);A3=A-ADDR        2040CC00 D
0703  A809 C0C1 AD90 0CFC          A2 C=A2,BMI;MR2;IF RDC                                           2041CC00 D
0704  B00C 00C0 0CC0 0030          COMP 1=SAR                 %MSB=GE;LSB=EQUAL BIT                 2042CC00 D
0705  4809 0C40 4000 00F0          B=A1                       %A1=Y                                 2043CC00 D
0706  AC08 E0C0 0C0C 00F0          WHEN RDC THEN A3=MAR,BEX   %B=Y+1;MAR=A                           2044CC00 D
0707  A809 0C40 1090 08F0          B=A3;MR1;IF RDC                                                  2045CC00 D
0708  AC08 0F46 0C0C 00F0          WHEN RDC THEN BMAR+1=MAR,BEX  %A1=Y;A3=Y+1;B=A                   2046CC00 D
0709  4809 2F56 000C 00F0          LIT AND BMAR=MAR           %CONVERT A(8) TO A(0)                 2047CC00 D
070A  A809 AC52 0000 C8F0          A1 EOL B;MR1;IF RDC                                              2048CC00 D
070B  6BC9 AC4C 0000 00F0          A1 XOR B; IF TRUE THEN SET LC1  %TEST SIGNS;LC1 => A EOL Y        2049CC00 D
C70C  0A50 C000 0000 00C0          IFETCH-1=AMPCR                                                   2050CC00 D
070D  480B AC5E 000C 40F0          A1 LSS B,CSAR; IF MST THEN STEP ELSE SKIP                        2051CC00 D
070E  7A49 00C0 0000 00F0          IF FALSE THEN SET LC2      %SIGNS WERE DIFFERENT                 2052CC00 D
070F  7249 0003 0090 00F0          IF TRUE THEN SET LC2                                             2053CC0C D
0710  AC08 00C0 0C90 C0F0          WHEN RDC THEN BEX                                                2054CC00 D
0711  4809 EC52 0000 00FC          A3 EOL B                                                         2055CC00 D
0712  640B C0CE 2000 C0F0          IF FALSE THEN A2 AND B110 = A2;STEP ELSE SKIP %CLEAR EQUAL BIT   2056CC00 D
C713  0F70 00C0 0090 0040          OPR05XX3-1=MPCR                                                  2057CC00 D
0714  2C0B C0DC 2090 00F0          IF LC1 THEN A2 OR B001 =A2;STEP ELSE SKIP %A+1,A=Y+1,Y           2058CC00 D
0715  482D D81D AD90 C0F0          A2 OR B100 C=A2;JUMP       %SET GEO BIT                          2059CC00 D
0716  3C19 D81C 2090 00F0          IF LC2 THEN A2 OR B100=A2;SKIP  %A+1,A GTR Y+1,Y SET GTR BIT     2060CC00 D
0717  4809 D80E 2090 00F0          A2 AND B011=A2             %CLEAR GEO BIT                        2061CC00 D
0718  482D C0CF A090 00F0          A2 AND B110 C=A2;JUMP      %CLEAR EQUAL BIT                      2062CC00 D
                                 OPR05XX3:  %LOWER HALVES A&Y NOT EQUAL                             2063CC00 D
0719  4809 EC4C 0C0C C0F0          A3 XOR B                                                         2064CC00 D
071A  4819 EC5E 0000 00FC          A3 LSS B;IF MST THEN SKIP                                        2065CC00 D
071B  7429 D81D A090 00F0          IF TRUE THEN A2 OR B100 C=A2;JUMP   %SET GEO BIT                 2066CC00 D
071C  7C29 D81D A090 C0F0          IF FALSE THEN A2 OR B100 C=A2;JUMP                               2067CC00 D
071D  482D D80F A090 00F0          A2 AND B011 C=A2;JUMP                                            2068CC00 D
                                 % ********************************     LOAD BASE & MEM.PROT.         ************2069CC00 D
                                 OPR05X4:                                                           2070CC00 D
071E  2240 0000 0090 C040          UNWRITTEN-1=MPCR           %PRINT ERROR AND RETURN TO IFETCH     2071CC00 D
                                 % ********************************     FLOATING POINT ADD            ************2072CC00 D
                                 OPR06X0:                                                           2073CC00 D
071F  4800 0000 0000 C0F0          WAIT                                                             2074CC00 D
0720  0A50 0C00 0090 0040          IFETCH-1=MPCR                                                    2075CC00 D
                                 % ********************************     FLOATING POINT SUBTRACT       ************2076CC00 D
                                 OPR06X1:                                                           2077CC00 D
0721  4800 0000 0000 00F0          WAIT                                                             2078CC00 D
0722  0A50 0000 0C00 C040          IFETCH-1=MPCR                                                    2079CC00 D
                                 % ********************************     FLOATING POINT MULTIPLY       ************2080CC00 D
                                 OPR06X2:                                                           2081CC00 D
0723  4800 0000 0000 00F0          WAIT                                                             2082CC00 D
0724  0A50 0000 0C00 0040          IFETCH-1=MPCR                                                    2083CC0C D
                                 % ********************************     FLOATING POINT DIVIDE         ************2084CC00 D
                                 OPR06X3:                                                           2085CC00 D
```

```
0725  4800 0000 0030 00FC          WAIT                                                                  2086CC0C D
0726  0A50 0000 0030 0040          IFETCH-1=MPCR                                                         2087C000 D
                                % ########################    FLOATING POINT ADD/ROUND     ##############2088CC00 D
                                OPR06X4:                                                                2089CC00 D
0727  4800 0000 0000 00F0          WAIT                                                                  2090CC00 D
0728  0A50 0000 0030 0040          IFETCH-1=MPCR                                                         2091CC00 D
                                % ########################    FLOATING POINT SUBR/ROUND    ##############2092CC00 D
                                OPR06X5:                                                                2093CC00 D
0729  4800 00C0 0000 00F0          WAIT                                                                  2094CC00 D
072A  0A50 0000 0000 0040          IFETCH-1=MPCR                                                         2095CC00 D
                                % ########################    FLOATING POINT MULT/ROUND    ##############2096CC00 D
                                OPR06X6:                                                                2097CC00 D
072B  4800 0000 00C0 00F0          WAIT                                                                  2098CC00 D
072C  0A50 0000 0000 0040          IFETCH-1=MPCR                                                         2099CC00 D
                                % ########################    FLOATING POINT DIV/ROUND     ##############2100CC00 D
                                OPR06X7:                                                                2101CC00 D
072D  4800 00C0 0000 00F0          WAIT                                                                  2102CC00 D
072E  0A50 0000 0000 0040          IFETCH-1=MPCR                                                         2103CC00 D
                                % ########################    INTERPROCESSOR INTERRUPT     ##############2104CC00 D
                                OPR07X0:                                                                2105CC00 D
072F  2240 0000 0000 004C          UNWRITTEN-1=MPCR          %PRINT ERROR AND RETURN TO IFETCH           2106CC00 D
                                % ########################    ALLOW ENABLE INTERUPT        ##############2107CC00 D
                                OPR07X1:                                                                2108CC00 D
0730  2240 0000 0000 0040          UNWRITTEN-1=MPCR          %PRINT ERROR AND RETURN TO IFETCH           2109CC00 D
                                % ########################    PREVENT ENABLE INTERRUPT     ##############2110CC00 D
                                OPR07X2:                                                                2111CC00 D
0731  2240 0000 0030 0040          UNWRITTEN-1=MPCR          %PRINT ERROR AND RETURN TO IFETCH           2112CC00 D
                                % ########################    LOAD IOC MONITOR CLOCK       ##############2113CC00 D
                                OPR07X3:                                                                2114CC00 D
0732  2240 0000 0000 0040          UNWRITTEN-1=MPCR          %PRINT ERROR AND RETURN TO IFETCH           2115CC00 D
                                % ########################      I/O ROUTINES      ###################### 2116CC00 D
                                OPR07X4:                                                                2117CC00 D
0733  4809 E012 0000 00F0          A3 EQL 0                                                              2118CC00 D
0734  680B 0000 0000 00F0          IF TRUE THEN STEP ELSE SKIP                                           2119CC00 D
0735  1F40 00C0 0030 0040          CARDREAD-1=MPCR                                                       2120CC00 D
0736  4809 E0D2 0000 00F0          A3 EQL 1                                                              2121CC00 D
0737  680B 0000 0030 00F0          IF TRUE THEN STEP ELSE SKIP                                           2122CC00 D
0738  1F80 0000 0030 0040          PRINTLINE-1=MPCR                                                      2123CC00 D
0739  1FC0 00C0 0000 0040          DISKREAD-1=MPCR                                                       2124CC00 D
                                % ########################    INTERRUPT RETURN             ##############2125CC00 D
                                OPR07X5:                                                                2126CC00 D
073A  2240 0000 0030 0040          UNWRITTEN-1=MPCR          %PRINT ERROR AND RETURN TO IFETCH           2127CC00 D
                                %   ####################      REPEAT      ################################ 2128CC00 D
                                OPR07X6:                    % RP                                        2129CC00 D
073B  48C9 0000 0038 00F0          LMAR;RESET GC2                                                        2130CC00 D
073C  00F0 00C0 0000 00E0          15 = LIT                                                              2131CC00 D
073D  A809 0000 0000 08F0          MR1; IF RDC               % B(7)                                      2132CC00 D
073E  AC08 0000 0C00 00F0          WHEN RDC THEN BEX                                                     2133CC00 D
073F  4809 0C42 0000 00F0          NOT B                                                                 2134CC00 D
0740  6C0B C0C0 2030 00F0          IF ABT THEN A2 + 1 = A2 ELSE SKIP                                     2135CC00 D
0741  0A50 00C0 0030 0040          IFETCH-1 = MPCR           % SKIP N.I.                                 2136CC00 D
0742  4809 0C40 1030 00F0          B = A3                                                                2137CC00 D
0743  4809 E0DE 0030 C0F0          A3-1 = MIR                                                            2138CC00 D
0744  2300 0000 0030 0060          OUTPUT1-1 = CPCR          % DEC B(7)                                  2139CC00 D
0745  4809 C003 001C 00FC          A2 = MAR2                                                             2140CC00 D
0746  A809 0000 0C00 CCF0          MR2; IF RDC                                                           2141CC00 D
0747  AC08 A000 0C98 00F0          WHEN RDC THEN A1=MIR,LMAR,BEX  %READ N.I.                             2142CC00 D
0748  0180 0000 0000 00E0          RINST=LIT                                                             2143CC00 D
0749  2300 0000 0030 C060          OUTPUT1-1 = CPCR          % SAVE REPEAT INST.                         2144CC00 D
074A  4809 0F46 000C C0F0          BMAR + 1 = MAR                                                        2145CC00 D
```

```
074B  4809 0C40 4090 C0F0          B = MIR,A1                                                    21460C00 D
074C  230C 00C0 0C00 0060          OUTPUT1-1 = CPCR          % SAVE N.I.                         21470C00 D
074D  4809 0C40 9000 00F0          B R = A3                                                      21480C00 D
074E  200C 00C3 0C00 0030          26 = SAR                                                      21490C00 D
074F  3C1C 0000 0C00 006C          CONDCHECK-1 = CPCR        % OPCODE REPEATABLE ?               21500C00 D
0750  2E99 C001 A00C 40F0          IF LC1 THEN A2 C = A2,CSAR;SET GC2; SKIP                      2151CC00 D
0751  0AAC 0C00 0C00 C040          IFETCH1-1 = MPCR          % NOT REPEATABLE                    2152CC00 D
0752  9000 0003 0000 0020          21 = SAR                                                      21530C00 C
0753  1888 0000 0000 00F0          SET GC2; WHEN GC2 THEN STEP                                   21540CC0 D
0754  4809 C0D0 A000 C0F0          A2 OR B001 C = A2         % SET REPEAT FLAG                   2155CC00 D
0755  300C 00C3 0090 0060          S6YFETCH-1 = CPCR                                             2156CC00 D
0756  0AAC 0000 0090 C040          IFETCH1-1 = MPCR                                              21570C00 D
                                % ********************      JUMP ON EVEN PARITY     ********** 21580C00 C
                                OPR50X0:    %A1=Y-ADDRESS; A3=A(A)                            21590C00 D
0757  A809 0C40 1090 08F0          B=A3;MR1;IF RDC           %READ  A(A+1) MAR=ADDRESS OF A+1    21600C00 D
0758  AC08 0000 0C90 00F0          WHEN RDC THEN BEX                                             21610C00 D
0759  4809 EC56 0B0C 00F0          A3 AND B=B                %A(A) AND A(A+1) = B                2162CC0C D
075A  292C 00C0 000C 0060          COUNTONES-1=CPCR          %COUNTONES USES A3 FOR COUNTER      21630C00 D
075B  4809 E000 0000 00F0          A3                        %CHECK LSB FOR ODD                  2164CC00 D
075C  5819 0000 0000 C0F0          IF LST THEN SKIP                                              21650C00 D
075D  23CB 0000 0090 00F0          IF NOT LC1 THEN SET LC1; STEP ELSE SKIP                       21660C00 D
075E  2BD9 0000 0090 00F0          IF LC1 THEN SET LC1; SKIP                                     21670C00 D
075F  0A50 0000 0C00 C040          IFETCH-1=MPCR                                                 21680C00 D
0760  4809 A0C0 1000 C0F0          A1=A3                     %PUT JUMP ADDRESS IN A3 FOR PUTJUMP 2169CC00 D
0761  2660 0000 0000 0040          PUTJUMP-1=MPCR                                                217C0C00 D
                                   %IF ODD(LST) AND NOT LC1(OPR5CX0) => NO JUMP                  21710C0C D
                                   %IF ODD(LST) AND LC1(OPR5CX1) => JUMP;SET LC1 FOR PUTJUMP     21720C00 D
                                % ********************      JUMP ON ODD PARITY  ********** 21730C00 D
                                OPR50X1:                    %USES OPR5CX0 ROUTINE WITH LC1 SET  21740C00 D
0762  49C9 0000 0000 C0F0          SET LC1                                                       2175CC00 D
0763  756C 0000 0090 0040          OPR50X0-1=MPCR                                                2176CC00 D
                                % ********************   JUMP IF A(A) AND A(A+1) EQUAL ZERO  ****** 2177CC00 D
                                OPR50X2:    %A1=JUMP ADDRESS;B=(A(A));MAR=A+1 ADDRESS          21780C00 D
0764  A809 0C52 0090 08F0          0 EQL B;MR1;IF RDC        %TEST A(A) FOR 0                    21790C00 D
0765  6019 00C0 0000 C0F0          IF FALSE THEN SKIP                                            218CCC00 D
0766  AC08 0000 0C90 00F0          WHEN RDC THEN BEX                                             21810C00 D
0767  4809 0C52 0000 00F0          0 EQL B                                                       21820C00 D
0768  6A49 0003 0000 00F0          IF TRUE THEN SET LC2      %TEST (A(A+1)=0                     21830C00 D
0769  3819 0000 0000 00F0          IF LC2 THEN SKIP          %A(A) & A(A+1)=0                    21840C00 D
076A  280B 0000 0000 00F0          IF LC1 THEN STEP ELSE SKIP                                    21850C00 D
076B  23D9 0000 009C 00FC          IF NOT LC1 THEN SET LC1; SKIP                                 21860C00 D
076C  0A5C 00C0 C090 C040          IFETCH-1=MPCR                                                 21870C00 D
076D  4809 A000 1000 00F0          A1=A3                     %SETUP JUMP ADDRESS FOR PUTJUMP     21880000 D
076E  2660 0000 0000 0040          PUTJUMP-1=MPCR                                                21890C00 D
                                   %IF A(A) AND A(A+1) = 0 AND OPR50X2 => JUMP                   21900C00 D
                                  %IF  A(A) AND A(A+1) = 0 AND OPR50X3 => NO JUMP                2191CC00 D
                                % ********************    JUMP IF A(A+ OR A(A+1) NOT EQUAL ZERO  21920C00 D
                                OPR50X3:    %USES SAME ROUTINES AS OPR50X2 WITH LC1 SET        21930C00 D
076F  49C9 0000 0000 00F0          SET LC1                                                       21940000 D
0770  7630 0000 0000 0040          OPR50X2-1=MPCR                                                2195CC00 D
                                % ********************      JUMP (A) POSITIVE       ********** 21960C00 D
                                OPR51X0:                    %B=(A)                              21970C00 D
0771  4809 0C40 0C90 00F0          B                                                             2198CC00 D
0772  43C9 0000 0090 00F0          IF NOT MST THEN SET LC1                                       21990C00 D
0773  2660 0000 0090 0040          PUTJUMP-1=MPCR                                                22000C00 D
                                % ********************       JUMP (A) NEGATIVE       ********** 22010C00 D
                                OPR51X1:                                                        22020C0C D
0774  4809 0C42 0B00 C0F0          NOT B=B                   %SAME AS 51X0 EXCEPT COMPLEMENT B   22030C00 D
0775  7700 0000 0090 0040          OPR51X0-1=MPCR                                                22040C00 D
                                % ********************     JUMP (A) ZERO       ********** 22050C00 D
```

```
0776  4809 0C42 0000 00F0          OPR51X2:                                                    22060C00 D
                                           NOT B                                             22070C00 D
0777  6BC9 0000 0C00 C0F0                   IF ABT THEN SET LC1                               2208CC00 D
0778  266C 0000 0C30 C040                   PUTJUMP-1=MPCR                                    22090C00 D
                                     % ********************      JUMP (A) NOT ZERO      ******************** 22100C00 D
                                     OPR51X3:                                                22110C00 D
0779  4809 0C42 0030 00F0                   NOT B                                             22120C00 D
077A  63C9 0000 0C30 C0F0                   IF NOT ABT THEN SET LC1                           22130C0C D
077B  2660 0003 0030 0040                   PUTJUMP-1=MPCR                                    2214CC00 D
                                     % ********************        LOAD E AND JUMP        ********** 22150C00 D
                                     OPR52X0:                        %JUMP ADDR. IN A3;A1=B-FIELD 22160C00 D
077C  4809 A012 0000 00F0                   A1 EQL 0                                          22170C00 D
077D  67D9 C000 0030 00F0                   IF FALSE THEN A2=MIR;SET LC1;SKIP  %MAR1=ADDR OF B(A) 22180C00 D
077E  0A50 00C0 0000 0040                   IFETCH-1=MPCR                                     22190C00 D
077F  2300 00C0 0000 C060                   OUTPUT1-1=CPCR                                    22200C00 D
0780  266C 0000 0000 0040                   PUTJUMP-1=MPCR                                    22210C00 D
                                     % ********************        INDEX JUMP B         ********** 2222CC00 D
                                     OPR52X1:                        %B=(B(A)); A1=P-FIELD     22230C00 D
0781  4809 A012 0030 C0FC                   A1 EQL 0                  %A3=JUMP ADDR;MAR1=ADDR OF B(A) 22240C00 D
0782  640B 0C42 0000 00F0                   IF FALSE THEN NOT B;STEP ELSE SKIP                22250C00 D
0783  67D9 0C40 403C C0F0                   IF NOT ABT THEN B=A1;SET LC1;SKIP  %NO JUMP IF B(A)=C 22260C00 D
0784  0A50 0000 0000 C040                   IFETCH-1=MPCR                                     22270C00 D
0785  4809 A0DE 0090 00F0                   A1-1=MIR                  %DECREMENT B(A) AND JUMP  22280C00 D
0786  2300 0000 0030 0060                   OUTPUT1-1=CPCR                                    22290C00 D
0787  266C 0000 0000 0040                   PUTJUMP-1=MPCR                                    22300C00 D
                                     % ********************      JUMP SY+B        ********************** 22310C00 D
                                     OPR52X2:                        %MIR=B-FIELD              22320C00 D
0788  4809 A001 1D00 00F0                   A1 L=A3,BMI               %JUMP TO SY+B             22330C00 D
0789  0080 0000 0000 00A0                   16=SAR;BBASE=LIT          %ISOLATE SY-FIELD         22340C00 D
078A  4809 2C40 003C C0F0                   LIT+B=MAR                 %MAR=ADDR OF B-REG        2235CC00 D
078B  A809 E0C0 9000 08F0                   A3 R=A3;MR1;IF RDC                                  22360C00 D
078C  AC08 00C0 0C30 00F0                   WHEN RDC THEN BEX         %B=(B)                    22370C00 D
078D  49C9 EC40 1000 00F0                   A3+B=A3;SET LC1           %A3=SY+B(B)               22380C00 D
078E  266C 0003 0000 0040                   PUTJUMP-1=MPCR                                      22390C00 D
                                     % ********************      UNCONDITIONAL JUMP LOWER ********** 22400C00 C
                                     OPR52X3:                        %A3=JUMP ADDRESS          22410C00 D
078F  4809 E003 0C1C 00F0                   A3=MAR2                                           22420C00 D
0790  A809 C000 A000 4CF0                   A2 R=A2,CSAR;MR2;IF RDC   %DELETE OLD PAR;READ JUMP ADDRESS 22430000 D
0791  83F0 0003 0000 00A0                   20=SAR;63=LIT             %LOWER WORD BIT 15 = LSB IS SET 2244CC00 D
0792  AC08 C0DD 2C30 00F0                   WHEN RDC THEN A2 OR B001 L=A2,BEX                   22450C00 D
0793  4809 0C41 C030 00F0                   B C=A1                    %LOWER WORD SHIFTED TO UPPER 2246CC00 D
0794  2000 0000 0000 0010                   10=SAR                    %SETUP OPCODE IO BE EXECUTED 22470000 D
0795  4809 A156 0C40 C0F0                   A1 AND LIT=AMPCR .        %OFCODE IN AMPCR          22480000 D
0796  4809 0C41 CC30 00FC                   B C=A1                    %SWAP LOWER WORD TO UPFER 22490C00 C
0797  0000 0000 000C 0020                   16=SAR                                            22500C00 D
0798  4824 CF5C 2C00 C0FC                   A2 OR BMAR=A2;EXEC        %INSERT NEW PAR           22510C00 D
0799  482D C0C0 2000 00F0                   A2+1=A2;JUMP                                      2252C000 D
                                     % ********************      JUMP OVERFLOW AND A        ********** 2253CC00 D
                                     OPR53X0:                        %B=A-FILED;A3=JUMP ADDRESS 22540000 D
079A  4809 0C52 0030 00F0                   0 EQL B                                           2255CC00 D
079B  63C9 C0C1 A000 40F0                   A2 C = A2,CSAR; IF FALSE THEN SET LC1              22560C00 D
079C  90C0 0000 C000 0030                   COMP 3=SAR                                        2257CC00 D
079D  266C 0000 003C 00C0                   PUTJUMP-1 = AMPCR                                 22580C00 D
079E  4809 C000 0030 00F0                   A2                                                2259CC00 D
079F  4819 D8CF A030 C0F0                   A2 AND B011 C = A2; IF MST THEN SKIP               2260CC00 D
07A0  23ED 00C0 0030 00F0                   IF NOT LC1 THEN SET LC1;JUMP ELSE JUMP             22610000 D
07A1  2BED 0000 0030 00F0                   IF LC1 THEN SET LC1;JUMP ELSE JUMP                 22620C00 D
                                           %JUMP IF A=0 AND NO OVERFLOW OR A=1 AND OVERFLOW    22630C00 D
                                     % ********************      JUMP ON CD EQUAL/UNEQUAL/LIMITS    **** 22640C00 D
                                     OPR53X1:                        %B= A-FILED; A3=JUMP ADDRESS 22650C00 D
```

```
07A2  4809 0C40 4000 C0F0          B = A1                                                    22660C00 D
07A3  4809 A640 0040 C0F0          A1+AMPCR=AMPCR              %OPCODE IN A1                 22670C0C D
07A4  3AD0 0000 0C00 C0C0          ACD-1=AMPCR                                               22680C00 D
07A5  4809 00C0 0000 C0F0          STEP                        % ALLOW AMPCR TO SET          22690C00 D
07A6  4824 C0C1 9000 C0F0          A2 C=A3;EXEC                %MSB=GE;LSB=EQL;2ND MSB=LIMITS BIT 22700C00 D
07A7  3000 0000 000C C030          31 = SAR                                                  22710C00 D
07A8  4836 E000 0000 00FC          A3;CALL                                                   2272CC00 D
07A9  49DB 0F40 1000 C0F0          BMAR = A3; SET LC1; SKIP    % RETURN HERE IF LEGAL JUMP   22730000 D
07AA  0A50 0000 0030 0040          IFETCH-1=MPCR                                             22740C00 D
07AB  266C 0000 0000 0040          PUTJUMP-1=MPCR                                            22750C00 D
                                 % ********************        RETURN JUMP        ********** 22760C00 D
                                 OPR53X2:                      %JUMP ADDRESS IN A3;B=A-FIELD 2277CC00 D
07AC  4809 0C40 4038 80F0          B=SAR,A1,LMAR               %A1=A-FIELD; MAR=SWITCHES     22780C00 D
07AD  0580 0000 0030 00E0          SWITCH =LIT                                               22790C00 D
07AE  A809 A15E 003C 08F0          A1 GEQ LIT;MR1;IF RDC                                     22800C00 D
07AF  0040 00C0 0000 00E0          4=LIT                       %A=4 THEN STOP                22810C00 D
07B0  7BC9 0000 0035 00F0          0=CTR; IF TRUE THEN SET LC1                               22820C00 D
C7B1  AC08 A002 0C00 00F0          WHEN RDC THEN NOT A1,BEX    %B=SWITCHES                   22830C00 D
07B2  6BC9 00C0 0C00 C0F0          IF ABT THEN SET LC1         %A=0 THEN JUMP                22840C00 D
07B3  4809 0C40 8B00 00F0          B R=B                       %CHECK PROPER BIT             22850C00 D
07B4  4809 0C40 0C30 00F0          B                                                         22860C00 D
07B5  5BC9 0000 0C00 00F0          IF LST THEN SET LC1         %SWITCH SET => JUMP           22870000 D
07B6  2FD9 A15E 000C 00F0          IF LC1 THEN A1 GEQ LIT;SET LC1;SKIP                       22880C00 D
07B7  0A5C 0003 0030 0040          IFETCH-1=MPCR                                             22890000 D
C7B8  7801 0000 0000 C0F0          IF TRUE THEN WAIT                                         22900000 D
07B9  4809 C001 4008 40F0          A2 L=A1,LMAR,CSAR           %SAVE THE PAR IN A1           2291CC00 D
07BA  810C 0000 0000 00A0          SBASE=LIT;COMP 12=SAR                                     22920C00 D
07BB  4812 A000 C030 C0F0          A1 R=A1,MIR;SAVE            %A1=0/PAR=>S&Y                22930C00 D
07BC  A809 0000 0030 08F0          MR1;IF RDC                  %READ BASE REG :X:            22940C00 D
07BD  AC08 0000 0C3C 00F0          WHEN RDC THEN BEX           %%FORM A+Y-(S) AND CHECK SIGN 22950C00 D
07BE  4809 AC5E 0D30 00F0          A1-B=MIR,BMI                %MIR =CURRENT DIFF.;B=PREV DIFF. 22960C00 D
07BF  4429 0F46 000E C0F0          IF NOT MST THEN BMAR+1=MAR,INC;JUMP                       2297CC0C D
07C0  4809 E003 001C 00F0          A3=MAR2                     %ADDRESS OF NEXT INST.        22980000 D
07C1  A809 6000 CC0C 0CF0          CIR R=A1;MR2;IF RDC         %A1=S/0                       22990C00 D
07C2  300C 0000 0C30 0010          11=SAR                                                    23000C00 D
07C3  AC08 AC5D 4C30 00F0          WHEN RDC THEN A1 OR B L=A1,BEX                            23010C00 D
07C4  0000 0000 0030 0020          COMP 16=SAR                                               23020C00 D
07C5  4809 0C40 8B30 00F0          B R=B                       %B=0/OPC,A,F,B                23030C00 D
07C6  4809 AC5D 8080 00F0          A1 OR B C=MIR               %OPC,A,F,B,S,Y                2304CC00 D
07C7  4809 E0C0 1000 00F0          A3 + 1=A3                   %A3=JUMP ADDRESS+1            2305CC00 D
C7C8  266C 0000 0030 00C0          PUTJUMP-1=AMPCR                                           23060C00 D
07C9  9809 0000 000C 1CF0          MW2;IF SAI                                                2307CC0C D
07CA  9C28 0000 0000 00F0          WHEN SAI THEN 0;JUMP                                      2308CC0C D
                                 % ********************        MANUAL JUMP        ********** 2309CC0C D
                                 OPR53X3:                      %B=A-FIELD;A3=JUMP ADDRESS    23100000 D
07CB  4809 0C40 4038 80FC          B=SAR,A1,LMAR               %SAR=A-FIELD                  2311CC00 D
07CC  058C 0000 0000 00E0          SWITCH =LIT                                               2312CC00 D
07CD  A809 A15E 0030 08F0          A1 GEQ LIT;MR1;IF RDC                                     2313CC00 D
07CE  0040 0000 0030 00E0          4=LIT                                                     2314CC0C D
07CF  7BC1 A002 0000 00F0          NOT A1; IF TRUE THEN WAIT;SET LC1                         23150C00 D
07D0  6BC9 0000 0030 00F0          IF ABT THEN  SET LC1        %LC1=>A EQL 0                 23160C00 D
07D1  AC08 00C0 0C30 00F0          WHEN RDC THEN BEX                                         2317CC00 D
07D2  4809 0C40 8B00 00F0          B R=B                       %B=SWITCH BIT IN LSB          23180000 D
07D3  4809 0C40 003C 00F0          B                                                         2319CC0C D
07D4  5BC9 0000 0C00 00F0          IF LST THEN SET LC1                                       23200C00 D
07D5  2BD9 0000 0000 C0F0          IF LC1 THEN SET LC1;SKIP                                  2321CC0C D
07D6  0A50 0000 0030 0040          IFETCH-1=MPCR                                             23220C00 D
07D7  2660 00C0 0000 0040          PUTJUMP-1=MPCR                                            23230C00 D
                                 % ********************     SCALE FACTOR                  **********2324CC0C D
                                 OPR70X0:                                                    23250C00 D
```

```
07D8  2250 0000 0030 0040          UNWRITTEN4-1=MPCR          %PRINT ERROR & RETURN TO HALFFETCH   23260C00 D
                                % ************************    DOUBLE SCALE FACTOR          ************2327CC00 D
                                OPR70X1:                                                            23280C00 D
07D9  225C 0000 0030 0040          UNWRITTEN4-1=MPCR          %PRINT ERROR & RETURN TO HALFFETCH   23290C00 D
                                % ************************    COMPLEMENT A                 ************23300C00 D
                                OPR70X2:                                                            23310000 D
07DA  4809 0C42 0030 00FC          NOT B=MIR                                                        2332CC00 D
07DB  0B8C 0000 0000 0040          HALFFETCH-1=MPCR                                                 23330C00 D
                                % ************************    DOUBLE COMPLEMENT A          ************23340C00 D
                                OPR70X3:    %B=(A);MAR=A-ADDRESS                                    23350C00 D
07DC  4809 0C42 0C30 00F0          NOT B=MIR                  %COMPLEMENT A                        2336C000 D
07DD  9809 00C0 0030 18F0          MW1; IF SAI                                                      2337CC00 D
07DE  9C08 0F46 000C 00F0          WHEN SAI THEN BMAR+1=MAR                                         2338CC00 D
07DF  4809 2F56 000C 00F0          LIT AND BMAR=MAR                                                 23390C00 D
C7E0  A809 0000 0030 C8F0          MR1;IF RDC                                                       23400C00 D
07E1  AC08 0000 0C30 00F0          WHEN RDC THEN BEX                                                2341CC00 D
07E2  4809 0C42 0030 00F0          NOT B=MIR                  %COMPLEMENT A+1                       23420C00 D
07E3  0B8C 0000 0030 0040          HALFFETCH-1=MPCR                                                 23430C00 D
                                % ************************    MULTIPLY REGISTERS A X A+1     **********  2344CC00 D
                                OPR74X0:   %B=A-FIELD;A3=B-FIELD                                    2345CC00 D
07E4  2270 0000 0030 C060          PUTPAR-1=CPCR              %SAVE A2                             23460C00 D
07E5  4809 E0C0 0C3C 00F0          A3=MAR                                                           23470C00 D
07E6  A809 A000 0030 08F0          A1=MIR;MR1;IF RDC          %READ A(B);CURRENT INSTR. IN MIR     23480C00 D
07E7  AC08 0C40 0C0C 00F0          WHEN RDC THEN B=MAR,BEX    %B=(A(B));MAR=A                      2349CC00 D
07E8  A809 0C40 2000 08F0          B=A2;MR1;IF RDC            %A2=MULTIPLIER                       235C0C0C D
07E9  AC08 0C00 0C30 C0F0          WHEN RDC THEN BEX          %B=MULTIPLICAND                      2351CC00 D
C7EA  2980 0000 0030 0060          MULTIPLY-1=CPCR                                                  2352CC00 D
07EB  4809 E000 0D30 00F0          A3=MIR,BMI                 %OUTPUT LEAST;SAVE OLD A2            23530C00 D
07EC  9809 0000 0030 18F0          MW1;IF SAI                                                       2354CC00 D
07ED  9C08 A0C0 0030 C0F0          WHEN SAI THEN A1=MIR                                             23550000 D
07EE  4809 0F46 000C 00F0          BMAR+1=MAR                 %OUTPUT MOST                         2356CC00 D
07EF  4809 2F56 0C0C 00F0          LIT AND BMAR=MAR           %CONVERT A(8) TO A(0)                2357CC00 D
07F0  9809 0C40 4000 18F0          B=A1;MW1;IF SAI            %RESET INSTRUCTION IN A1             2358CC00 D
07F1  0BC0 0000 0000 00C0          HALFFETCH=AMPCR            %PREPAR EXIT TO HALFFETCH +1         2359CC00 D
07F2  9C08 0000 0030 00F0          WHEN SAI THEN 0;STEP                                             2360CC00 D
C7F3  22A0 0000 0030 0040          GETPAR-1=MPCR              %RESTORE A2;RETURN TO HALFFETCH      2361CC00 D
                                % ************************  DIVIDE A(A+1),A(A) / A(B) => A(A)       23620000 D
                                OPR74X1:   %B=A-FIELD; A3=B-FIELD                                   2363CC00 D
C7F4  2270 0000 0000 0060          PUTPAR-1=CPCR              %SAVE A2                             23640C00 D
07F5  4809 0C46 000C 00F0          B+1=MAR                    %READ A+1                            23650C00 D
07F6  4809 2F56 000C C0F0          LIT AND BMAR=MAR           %CONVERT A(8) TO A(0)                23660C00 D
C7F7  0070 0000 0000 00E0          7=LIT                                                            23670C00 D
07F8  A809 A000 0030 08F0          A1=MIR;MR1;IF RDC          %SAVE A1                             23680C00 D
07F9  4809 0C40 2030 C0F0          B=A2                       %SAVE A-ADDRESS                      23690C00 D
07FA  AC08 E000 0C0C 00F0          WHEN RDC THEN A3=MAR,BEX                                         23700C00 D
07FB  A809 0C40 1000 C8FC          B=A3;MR1;IF RDC                                                  23710000 D
07FC  AC08 C000 0C0C 00F0          WHEN RDC THEN A2=MAR,BEX   %A-ADDRESS IN MAR                    23720C00 D
07FD  2A80 00C0 0030 0060          DIVIDE-1=CPCR              %A3=MOST DIVIDEND;B=DIVISOR          23730000 D
07FE  4809 C0C0 0D30 00F0          A2=MIR,BMI                 %A2=QUOTIENT;A1=REMAINDER            2374CC00 D
07FF  6C09 00C0 003C 00FC          IF ABT THEN 0=MIR                                                23750C00 D
0800  9809 0000 0030 18F0          MW1;IF SAI                                                       2376CC00 D
0801  9C08 0F46 000C 00F0          WHEN SAI THEN BMAR+1=MAR                                         2377C000 D
0802  4809 2F56 000C 00F0          LIT AND BMAR=MAR           %CONVERT A(8) TO A(0)                2378CC00 D
0803  0070 00C0 0030 00E0          7=LIT                                                            23790C00 D
C804  4809 A0C0 0030 C0F0          A1=MIR                                                           23800C00 D
0805  6C09 0C00 0C30 C0F0          IF ABT THEN 0=MIR                                                2381CC00 D
0806  9809 0C40 4000 18F0          B=A1; MW1; IF SAI          %RESET INSTRUCTION IN A1             2382CC00 D
0807  9C08 0000 0000 00F0          WHEN SAI THEN 0;STEP                                             23830C00 D
0808  22A0 00C0 0030 C060          GETPAR-1=CPCR                                                    23840C00 D
0809  4809 C0C1 A000 40F0          A2 C=A2,CSAR                                                     23850C00 D
```

```
080A  9000 0000 0000 0030          COMP 3=SAR                                                    23860000 D
0808  2C19 D81D A000 00F0          IF LC1 THEN A2 OR B100 C=A2;SKIP                              23870C00 D
080C  4809 D80F A000 00F0          A2 AND B011 C=A2            %CLEAR OVERFLOW                   23880C00 D
080D  0BCC 0000 0000 0040          HALFFETCH=MPCR               %ENTER HALFFETCH AT +1           23890C00 D
                                % ********************     SQUARE ROOT                 ********23900C00 D
                                OPR74X2:    % A3=B-FIELD; B=A-FIELD                             2391CC00 D
080E  2270 0000 0000 0060          PUTPAR-1=CPCR               %SAVE A2                          23920C00 D
080F  4809 A000 0058 00F0          A1=MIR,LMAR                 %SAVE CURRENT INSTRUCTION         23930C00 D
0810  0190 0000 0000 00B0          CINST=LIT;COMP 8=SAR                                          2394CC00 D
0811  9809 E001 0010 18F0          A3 L=BR2;MW1;IF SAI         %SAVE B-FIELD IN BR2              23950C00 D
0812  9C08 0C46 000C C0F0          WHEN SAI THEN B+1=MAR       %SETUP FOR A+1                    23960C00 D
0813  4809 2F56 000C 00F0          BMAR AND LIT=MAR            %CONVERT A(8) TO A(0)             23970C00 D
0814  0070 0000 0030 00E0          7=LIT                                                         2398CC00 D
0815  A809 0000 0000 C8F0          MR1;IF RDC                  %READ A+1                         2399CC00 D
0816  AC08 0C40 0C0C 00F0          WHEN RDC THEN B=MAR,BEX     %B=(A+1);MAR=A                    24000C00 D
0817  A809 0C40 000C C8FC          B;MR1;IF RDC                %READ A                           24010C00 D
0818  4E4B 00C6 0B30 00F0          IF MST THEN BC01+1=B;SET LC2;STEP ELSE SKIP                    24020C00 D
0819  2130 0000 0000 0040          ERRORMSG-1=MPCR             %SQUARE ROOT OF NEGATIVE          24030C00 D
081A  4809 0C40 4001 00F0          B=A1,LCTR                                                     24040000 D
081B  A0F0 0000 0030 00B0          15=LIT;COMP 2=SAR           %A1=(A+1)                         24050C00 D
081C  4852 0000 3B30 00F0          0=A3,A2,B,MIR;SET LC2;SAVE %INITIALIZE REGISTERS              24060C00 D
081D  4809 A000 8B02 C0F0    SQRT1:  A1 R=B,INC                                                 2407CC00 D
081E  4809 CC5C 2D30 00F0          A2 OR B=A2,EMI              %OR 2 MS BITS INTO A2             2408CC00 D
081F  4809 C012 0C00 00F0          A2 EOL 0                                                      24090C00 D
0820  6419 CC58 0000 00F0          IF FALSE THEN A2 GTR B;SKIP                                   24100C00 D
0821  0F80 00C0 0C0C C040          SQRT2-1=MPCR                                                  24110C00 D
0822  7C19 0C46 0B00 C0FC          IF TRUE THEN B+1=B;SKIP                                       24120C0C D
0823  0F90 00C0 0030 0040          SQRT2-1=MPCR                                                  2413CC00 D
0824  4809 CC5E 2C30 C0F0          A2-B=A2                                                       24140C00 D
0825  4809 E0DC 103C 00F0          A3 OR B001=A3                                                 24150C00 D
0826  8419 0C40 0000 C0F0    SQRT2:  IF NOT CRY THEN B;SKIP                                     2416CC00 D
0827  0FAC 00C0 0030 C040          SQRT3-1=MPCR                                                  2417CC00 D
0828  5C09 0C46 0B00 00F0          IF LST THEN B+1=B                                             24180C00 D
0829  4809 0C41 0030 00F0          B L=MIR                     %DOUBLE B                         24190000 D
082A  B000 C000 0C30 0030          COMP 1=SAR                                                    24200C00 D
082B  4809 E0CA 103C 00F0          A3 DAD 0=A3                                                   24210C00 D
082C  4809 A0C1 4000 00F0          A1 L=A1                                                       2422CC00 D
082D  A000 0000 0030 003C          COMP 2=SAR                                                    24230C00 D
082E  482D C001 2C30 00F0          A2 L=A2;JUMP                                                  24240C00 D
082F  3C19 E00A 103C C0F0    SQRT3:  IF LC2 THEN A3 DAD 0=A3;SKIP                               2425CC00 D
0830  0FB0 0000 0000 C04C          SQRT4-1=MPCR                %A3=ANSWER;A2=RESIDUE             24260C00 D
0831  AC08 C001 2C01 00F0          WHEN RDC THEN A2 L=A2,BEX,LCTR                                24270C00 D
0832  482D 0C40 4000 24F0          B=A1;ASE;JUMP               %SELECT BR2                       24280C00 D
0833  4809 E000 0C30 24FC    SQRT4:  A3=MIR;ASE                  %SELECT BR2                       24290C00 D
0834  4809 0F40 800C 00F0          BMAR R=MAR                  %SETUP FOR A(B)                   24300C00 D
0835  007C 00C0 0030 C090          8=SAR;7=LIT                                                   24310000 D
0836  9809 0C00 0000 18F0          MW1;IF SAI                  %OUTPUT ANSWER                    24320C00 D
0837  9C08 0F46 000C 00F0          WHEN SAI THEN BMAR+1=MAR                                      24330C00 D
0838  4809 2F56 000C C0F0          BMAR AND LIT=MAR            %CONVERT A(8) TO A(0)             24340C00 D
0839  4809 C000 0030 C0F0          A2 = MIR                                                      2435CC0C D
083A  9809 0000 0000 18F0          MW1;IF SAI                  %OUTPUT RESIDUE                   2436CC0C D
083B  9C08 0000 0030 00F0          WHEN SAI THEN 0;STEP                                          24370C00 D
083C  22A0 0000 0030 0060          GETPAR-1=CPCR               %RESTORE A2 TO PAR'               24380C00 D
083D  4809 000C 0058 00F0          LMAR                        %FETCH CURRENT INSTRUCTION        24390C00 D
083E  0190 00C0 003C 00EC          CINST=LIT                                                     2440CC00 D
083F  A809 0000 0000 08F0          MR1;IF RDC                                                    2441CC00 D
0840  AC08 00C0 0C0C 00F0          WHEN RDC THEN BEX                                             24420C00 D
0841  4809 0C40 4000 C0F0          B=A1                        %RESTORE A1 TO CURRENT INSTRUCTION 2443CC00 D
0842  0BCC 0000 C030 0040          HALFFETCH=MPCR              %ENTER HALFFETCH AT +1            24440000 D
                                % ********************     LOAD B(A) WITH B(B)       ********** 24450C00 D
```

```
0843  4809 E140 003C 00F0          OPR74X3:    %MIR=A-FIELD; A3=B-FIELD                                  24460C0C D
                                           A3+LIT=MAR                  %MAR=B(B)                         24470C00 D
0844  0080 0000 0030 00E0                  BBASE=LIT                                                     24480C00 D
0845  A809 0000 0000 08FC                  MR1; IF RDC                                                   24490C00 D
0846  AC08 0000 0C00 00F0                  WHEN RDC THEN BEX           %READ B(B)                        24500C00 D
0847  4809 0C40 0D90 00F0                  B=MIR,BMI                                                     24510C00 D
0848  4809 0C52 0000 00F0                  0 EQL B                     %CHECK FOR B(0) => ILLEGAL        24520C00 D
0849  640B 2C40 003C 00F0                  IF FALSE THEN LIT + B=MAR;STEP ELSE SKIP                      24530C00 D
084A  0BB0 0003 0000 C040                  HALFFETCH-1=MPCR            %MAR=B(A) ADDRESS                 24540C00 D
084B  0BCC 0000 0030 0040                  HALFFETCH=MPCR              %ENTER AT +1 AND DO NOT WRITE B(0) 24550C00 D
                                         % ********************      COMPARE REGISTERS    ****************** 24560C0C D
                                         OPR74X4:     %A3=A(B); B=A(A)                                    24570000 D
084C  4809 E0C0 003C 00F0                  A3=MAR                      %READ A(B)                        24580C00 D
084D  A809 0000 0030 08F0                  MR1; IF RDC                                                   24590C00 D
084E  AC08 0C00 0C00 00F0                  WHEN RDC THEN BEX                                             24600C00 D
084F  4809 0C40 1D00 00F0                  B=A3,BMI                    %A3=(A(B))                        24610C00 D
0850  4809 0C40 003C C0F0                  B=MAR                       %MAR = ADDRESS OF A(A)            24620C00 D
0851  A809 00C0 0000 08F0                  MR1; IF RDC                                                   24630C00 D
0852  AC08 0000 0C00 00F0                  WHEN RDC THEN BEX           %A3=A(E);B=A(A)                   24640C00 D
0853  2860 00C0 0030 0040                  SETCOMPARE-1=MPCR                                             24650C00 D
                                         % ********************    COMPARE A(B) WITH A(A+1); A(A) SET CD****** 24660C00 D
                                         OPR74X5:     %B,MIR=A-FIELD; A3=B-FIELD                          24670000 D
0854  4809 0C40 003C 20F0                  B=MAR;ASR                   %SETUP FOR BMAR1                  24680C00 D
0855  0BC0 0000 0C30 00C0                  HALFFETCH=AMPCR                                               24690C00 D
0856  A809 0000 0000 08F0                  MR1; IF RDC                                                   24700C00 D
0857  AC08 0F46 0C0C C0F0                  WHEN RDC THEN BMAR+1=MAR,BEX                                  24710C00 D
0858  4809 2F56 000C 00F0                  LIT AND BMAR=MAR            %CONVERT A(8) TO A(0)             24720C00 D
0859  A809 0C40 0090 08F0                  B=MIR;MR1; IF RDC           %MIR=(A(A))                       24730000 D
085A  AC08 E0C0 0C0C 00F0                  WHEN RDC THEN A3=MAR,BEX    %B=(A(A+1))                       24740C00 D
085B  A809 0C40 1000 08F0                  B=A3;MR1; IF RDC            %A3=(A(A+1))                      24750C00 D
085C  AC08 00C0 0C30 00F0                  WHEN RDC THEN BEX           %B=(A(B))                         24760C00 D
                                         OPR74XX5: %MIR=A(A);A3=A(A+1);B=A(B)                            24770C00 D
085D  4809 C0C1 A000 40F0                  A2 C=A2,CSAR                %LIMITS BIT IS MSB                24780000 D
085E  A000 0000 0030 0030                  COMP 2=SAR                                                    24790C00 D
085F  4809 EC4C 0C00 00F0                  A3 XOR B                    %TEST DIFFERENT SIGNS             24800C00 D
0860  480B EC58 0000 C0F0                  A3 LEQ B; IF MST THEN STEP ELSE SKIP                          24810C00 D
0861  7C2B D81D A000 C0F0                  IF FALSE THEN A2 OR B100 C=A2;JUMP ELSE SKIP                  24820C0C D
0862  7429 D81D AC00 00F0                  IF TRUE THEN A2 OR B100 C=A2;JUMP  %SAME SIGNS                24830C00 D
0863  4809 0C40 1D00 C0F0                  B=A3,BMI                                                      24840000 D
0864  4809 EC4C 0000 00F0                  A3 XOR B                    %A3=A(B);B=A(A)                   24850000 D
0865  480B EC5E 0000 00F0                  A3 LSS B; IF MST THEN STEP ELSE SKIP                          24860000 D
0866  7C2B D81D A000 00F0                  IF FALSE THEN A2 OR B100 C=A2;JUMP ELSE SKIP                  24870000 D
0867  7429 D81D AC00 C0F0                  IF TRUE THEN A2 OR B100 C=A2;JUMP                             24880C00 D
0868  482D D80F A000 C0F0                  A2 AND B011 C=A2;JUMP                                         24890C00 D
                                         % ********************    COMPARE A(A+1) AND A(A) WITH A(B)    ****** 24900C00 D
                                         OPR74X6:     %B,MIR=A-FIELD; A3=B-FIELD                          24910C00 D
0869  4809 0C40 0C3C 20F0                  B=MAR;ASR                   %SETUP FOR A(A); BMAR=MAR1        24920C00 D
086A  A809 0000 0000 08F0                  MR1; IF RDC                                                   24930C0C D
086B  AC08 0F46 0C3C C0F0                  WHEN RDC THEN BMAR+1=MAR,BEX                                  24940C00 D
086C  4809 2F56 000C 00F0                  LIT AND BMAR=MAR            %CONVERT A(8) TO A(0)             24950C00 D
086D  A809 0C40 0090 C8F0                  B=MIR;MR1; IF RDC           %B=(A(A))                         24960C00 D
086E  AC08 E000 0C0C 00F0                  WHEN RDC THEN A3=MAR,BEX                                      24970C0C D
086F  A809 0C40 1D00 08F0                  B=A3,BMI;MR1; IF RDC        %B=(A(A));A3=(A(A+1))             24980000 D
0870  AC08 EC56 1C30 00F0                  WHEN RDC THEN A3 AND B=A3,BEX                                 24990C00 D
0871  2860 0000 0000 004C                  SETCOMPARE-1=MPCR           %A3=A(B)                          25000C00 D
                                         %********************      COMPARE B(B) WITH B(A)    ************** 25010C00 D
                                         OPR74X7:     %B,MIR=A-FIELD; A3=B-FIELD                          25020C00 D
0872  4809 2C40 003C C0F0                  LIT+B=MAR                                                     25030C00 D
0873  0080 0000 0030 00E0                  BBASE=LIT                                                     25040C00 D
0874  A809 0000 0000 C8F0                  MR1; IF RDC                 %READ B(A)                        25050C00 D
```

```
0875  AC08 E140 0C0C C0F0          WHEN RDC THEN A3+LIT=MAR,BEX                                          25060C00 D
0876  A809 0C40 009C C8F0          B=MIR;MR1;IF RDC                                                      25070C00 D
0877  AC08 0000 0C0C 00F0          WHEN RDC THEN BEX                                                     25080C00 D
C878  4809 0C40 1D00 00F0          B=A3,BMI                    %B=B(A);A3=B(B)                           25090C00 D
0879  286C 0000 0000 0040          SETCOMPARE-1=MPCR           %B=B(B);A3=B(A)                           25100C00 D
                                % ************************    STORE IOC MONITOR CLOCK      ************25110C00 D
                                OPR77X0:                                                                25120C00 D
087A  2250 0000 0000 0040          UNWRITTEN4-1=MPCR           %PRINT ERROR & RETURN TO HALFFETCH        25130C00 D
                                % ************************    STORE REAL-TIME CLOCK        ************25140C00 D
                                OPR77X1:                                                                25150C00 D
087B  2250 00C0 0C00 0040          UNWRITTEN4-1=MPCR           %PRINT ERROR & RETURN TO HALFFETCH        25160C00 D
                                % ************************    PREVENT CLASS 3 INTERRUPT    ************25170C00 D
                                OPR77X4:                                                                25180C00 D
087C  2250 0000 0000 0040          UNWRITTEN4-1=MPCR           %PRINT ERROR & RETURN TO HALFFETCH        25190C00 D
                                % ************************    ALLOW CLASS 3 INTERRUPT      ************25200C00 D
                                OPR77X5:                                                                25210C00 D
087D  2250 0000 0000 0040          UNWRITTEN4-1=MPCR           %PRINT ERROR & RETURN TO HALFFETCH        25220C00 D
                                % ************************    STOP PROCESSOR/WAIT INT.     ************25230C00 D
                                OPR77X6:                                                                25240C00 D
087E  4800 0000 0000 00F0          WAIT                                                                  25250C00 D
087F  4809 0018 001C 00F0          B111=MAR2                   %WRITE TO MAILBOX                         25260C00 D
0880  4809 20C1 0090 C0F0          LIT L=MIR                   %SIGNALS LOADER TO RESTART                25270C00 D
0881  0080 0000 0000 C0A0          8=LIT;16=SAR                                                          25280C00 D
C882  7809 00C0 0000 1CF0          MW2;IF SAI                                                            25290C00 D
0883  7F48 0000 0001 C0F0          WHEN SAI THEN RESET GC1;LCTR %GIVE GC1 TO LOADER                      25300C00 D
0884  0640 0000 0C00 C0E0          1C0=LIT                     %WASTE TIME                               25310C00 D
0885  8808 0000 0002 00F0          INC; WHEN COV THEN STEP                                               25320C00 D
C886  0A00 0000 0C00 0040          START=MPCR                  %RESTART PROGRAM                          25330C00 D
0887  0000 00C0 0000 0040          END                                                                   25340C00 D
0830  832C 0000 000C 0040       ?END
0827  82E0 0000 0000 0040
0823  8250 00C0 0000 0040
0821  825C C0C0 0000 0040
0713  7180 0000 0000 0040
06CB  6CD0 0000 0000 C040
064A  65B0 0000 0000 00C0
062A  6340 0000 0000 0040
0610  618C 0000 000C 0040
05FB  600C 0000 000C 0040
0570  95C0 0000 0000 0040
04C2  9C80 0000 0000 0040
0476  5100 00C0 0C00 C040
0446  6C70 0000 0000 0040
03D6  41C0 00C0 0000 C0C0
03CE  3D40 0000 0000 00C0
03CA  3CD0 0000 0000 00C0
03C5  3DC0 0000 000C C0C0
03B5  3BF0 0CC0 0000 00C0
03B4  3BD0 0000 0000 00CC
03B3  3BB0 00C0 0000 00C0
03B2  3BA0 0000 0C00 00C0
03B1  3B90 0000 0000 00C0
03B0  3B7C 0000 0000 00CC
03AF  3B60 0000 0C00 00C0
03AE  3B5C 0000 000C C0C0
03A7  3AD0 0000 0C00 00C0
0397  3A40 0000 0C00 C0C0
0396  3A20 00C0 0000 00C0
0395  3A00 0000 0000 00C0
0394  39F0 0000 0000 00C0
```

```
0393  39D0 0000 0000 00C0
0392  39B0 0000 0000 00C0
0391  399C 0000 0000 C0C0
0390  3970 0000 C000 00C0
0389  38F0 0000 0000 00C0
0380  3C1C 0000 0000 C060
0377  37DC 0CC0 0000 C040
0366  3720 0000 0000 0040
035C  361C C000 0C00 C040
0349  354C 0000 CC00 0040
02E5  3860 0C00 0000 C040
C2E3  3A5C 0000 0000 00C0
02CF  2D3C 0000 0000 0060
02C6  2C80 0000 0000 0040
02BC  2C3C 0000 0000 0C40
02B6  2BAC 0000 0000 C040
022A  230C 0000 0000 0040
01FC  209C 0000 0000 0040
01F8  2090 00C0 0000 0040
01F4  20AC 0000 0000 C040
01F0  20A0 00C0 0C00 0040
01EE  2320 0000 0000 C040
01E7  2320 0000 0000 0040
01DF  2320 0C00 0000 0040
01D7  2320 0000 0000 C040
01CF  2320 0000 0C00 0040
01CD  2320 0000 0000 0040
01C6  2320 0000 0000 0040
01BE  1E7C 0000 0000 00C0
01BD  1DF0 0000 0000 00C0
01BC  1D70 0000 0000 00C0
01BB  1CF0 00C0 0000 00C0
01BA  1CDC 00C0 000C 00C0
01B9  1C60 0000 0C00 C0C0
C1B8  1BE0 0000 0000 C0C0
0187  1970 0000 0000 C040
0176  17E0 0000 0000 0060
0175  1770 0000 0000 0040
0172  17A0 0000 0C00 C040
0161  16C0 0000 0000 C04C
015E  186C 00C0 0000 00C0
0144  144C 0000 0000 0040
C129  137C 0000 0000 C040
0126  1440 00C0 000C 0040
0CFB  1150 00C0 0C0C 00C0
0CFA  111C 0000 0000 00C0
0DF9  10D0 C000 0C00 C0C0
0CF8  10A0 0000 0000 00C0
0CF7  108C 00C0 0000 00C0
0CF6  1020 C0C0 0000 00C0
0DF5  0FB0 0000 0C00 00C0
0CED  0F30 0C00 0C0C 00C0
0CD8  0DF0 0000 0000 C040
0CD4  1190 0000 0000 0040
0DCA  1EE0 00C0 0000 0060
0CC8  2270 0000 0C00 006C
0CB3  3370 00C0 0000 0060
0DAA  2CAC 00C0 0C00 C060
0CA8  1EE0 00C0 0C00 0060
0CA6  2270 0000 0000 0060
```

```
0CA2  22A0 0000 0000 0060
0C9F  21E0 0000 0C00 0040
0C9E  87D0 C0C0 0000 00C0
0C9D  87C0 0000 0C00 00C0
0C9C  87B0 0CC0 0000 C0C0
0C9B  21E0 0000 0C00 C040
0C9A  21E0 00C0 0000 0040
0C99  87A0 0000 0C00 00C0
0C98  8790 00C0 0000 00C0
0C97  8710 0000 0000 00C0
0C96  8680 0000 0000 C0C0
0C95  8530 00C0 0000 C0C0
0C94  84B0 0000 0000 C0C0
0C93  8420 0000 0000 C0C0
0C92  80D0 00C0 000C 00C0
0C91  7F30 0000 0000. C0C0
0C90  7E30 0000 0000 00C0
0C8F  21E0 00C0 0000 C040
0C8E  21E0 0000 0000 0040
0C8D  21E0 0000 0000 0040
0C8C  21E0 0000 0000 0040
0C8B  7DB0 0000 0000 C0CC
C08A  7D90 0000 0000 00CC
0C89  7D80 00C0 000C C0C0
0C88  7D70 0000 0C00 00C0
0087  21D0 0000 C000 00C0
0086  21D0 0000 0000 00C0
0C85  21D0 0000 0000 00C0
0C84  7CA0 0000 0000 00C0
0C83  7AB0 0000 0C00 00C0
0C82  7A10 0000 0000 00C0
0081  7990 0000 0000 00C0
0C80  21D0 0000 0000 C0C0
0C7F  21D0 0000 000C 00C0
0C7E  21D0 0000 0000 00C0
0C7D  21D0 0000 0000 C0CC
0C7C  78EC 0000 0000 00C0
0C7B  7870 00C0 0000 00C0
0C7A  7800 0000 0000 00C0
0C79  77B0 C000 0000 00C0
0C78  21DC 00C0 0000 00C0
0C77  21D0 0000 0C00 00C0
0C76  21D0 0000 0000 00C0
0C75  21D0 00C0 0000 00C0
0C74  7780 0000 0C00 C0C0
0C73  7750 0000 000C 00C0
0C72  7730 0000 0000 00C0
0C71  7700 0000 0C00 C0C0
0C70  21DC 000C C000 C0C0
0C6F  21DC 0000 0000 00CC
0C6E  21DC 0000 0000 00C0
0C6D  21DC 0000 0000 00C0
0C6C  76E0 0000 0000 00C0
0C6B  7630 0000 0000 00C0
0C6A  7610 0000 0000 00C0
0C69  7560 0000 0C00 00C0
0068  21D0 0000 0000 00C0
0C67  73AC 0000 0C00 00C0
0C66  7390 0000 0C00 00C0
0065  7320 0000 0000 C0C0
```

```
0C64  7310 0000 0C00 00C0
0C63  730C 00C0 C000 C0C0
CC62  72F0 C0C0 0000 00CC
0C61  72E0 0000 0C00 C0C0
0C60  72CC 0000 0000 00C0
0C5F  72A0 0000 0000 00C0
0C5E  7280 0000 0000 C0C0
0C5D  7260 0000 000C 00C0
0C5C  7240 0000 0C00 C0C0
0C5B  7220 0000 0000 00CC
0C5A  7200 00C0 0C00 00C0
CC59  71EC 00C0 0000 00C0
0C58  21D0 0000 0000 C0C0
CC57  21D0 00C0 0000 00CC
0C56  21D0 00C0 0000 00C0
0C55  71D0 0000 0C00 00C0
0C54  701C 0000 0000 C0CC
C053  5ED0 0000 0C00 00C0
0C52  5D70 0000 0C00 00C0
0051  5CFC 00C0 0C00 00C0
CC50  5BFC 0000 0000 00C0
0C4F  5B30 0000 0000 00CC
0C4E  5A80 0000 0C00 00C0
0C4D  5A10 00C0 0000 C0C0
0C4C  5990 0000 0C00 C0C0
0C4B  5970 0000 0C00 C0C0
0C4A  21D0 0000 0000 00C0
0C49  5940 0000 C000 00C0
0C48  5900 0000 0000 00CC
0C47  5880 00C0 0000 C0CC
0C46  5860 00C0 0000 00C0
0C45  57E0 0000 0000 00C0
0C44  57C0 0000 0000 00CC
0043  5740 0000 0C00 00C0
0C42  572C 0000 0000 C0C0
0C41  5700 0000 0000 00C0
0C40  56AC 0000 0000 00C0
0C3F  21EC 0000 0000 C040
0C3E  21EC 0000 0000 0040
CC3D  5610 0000 CC00 C0C0
0C3C  21EC 0000 0000 0040
0C3B  21E0 0C00 0000 0040
0C3A  547C 0000 0000 00C0
0C39  540C 0C00 000C C0C0
0C38  5240 0C00 0000 00C0
0C37  51F0 00C0 0C00 00C0
0C36  50A0 0000 0C00 00C0
0C35  5070 0000 0000 00C0
0C34  4EF0 0000 0000 00C0
CC33  4EB0 00C0 0000 00C0
0C32  4DCC 0000 0000 00C0
0C31  4CD0 C0C0 0000 C0C0
CC30  4C90 0000 000C 00C0
0C2F  4BCC 0C00 0000 C0C0
0C2E  4B80 00C0 000C 00C0
0C2D  4AD0 0000 0000 C0C0
CC2C  4A50 0000 0000 00C0
0C2B  4990 0000 0000 00C0
0C2A  48F0 0000 0000 C0C0
0C29  489C 0000 0C00 00C0
```

```
0028  5790 0000 0000 C0C0
0027  5700 0000 0000 00C0
0026  565C 0000 0000 00C0
0025  55EC 00C0 0000 00C0
0024  54B0 0000 0000 00C0
0023  5410 0000 000C C0CC
0022  527C 0000 000C 00CC
0021  519C 0000 0000 00C0
0020  5130 0000 0C00 C0C0
001F  50E0 0CC0 0000 00C0
001E  5070 00C0 0000 00CC
001D  4FDC 0000 0000 00C0
001C  4FB0 0C00 000C C0C0
001B  4F10 00C0 0C00 C0C0
001A  21D0 00C0 0000 00C0
0019  21DC 00C0 000C 00C0
0018  4EA0 0000 0C00 C0C0
0017  4E50 0000 000C 00C0
0016  4CF0 0000 0000 00C0
0015  4CA0 0000 0000 00C0
0014  4BDC 0000 0000 C0C0
0013  4BD0 0000 0C00 C0C0
0012  4A30 0000 0000 00C0
0011  49BC 00D0 0000 00C0
0010  496C 00C0 0000 00C0
000F  4920 0000 000C 00C0
000E  487C 0000 0000 00C0
000D  47F0 0D00 0000 00C0
000C  477D 0000 0000 00C0
000B  46B0 0000 0C00 00C0
000A  45F0 0000 00D0 00C0
0009  45C0 0000 0C00 00C0
0008  453C 0000 0000 C0C0
0007  452C 00C0 0C00 00C0
0006  44A0 0C00 CC00 C0C0
0005  21D0 C0C0 000C C0C0
0004  4400 00C0 0000 C0C0
0003  4360 0000 0000 00C0
0002  42C0 0000 0000 00C0
0001  21D0 0D00 000C 00C0
0000  D9F0 C000 000C 0040
 0 ERRORS    2536 CARDS.  14930 TOKENS.     26338 RULES.      1792 SECONDS.
 0 WARNINGS.      100 DISK SECTORS.
```

APPENDIX C. LOADER PROGRAM LISTING

This appendix provides a copy of the Microtranslator output listing of the Loader written in conjunction with the Emulator. Its source is maintained on disk and cards; and its object module is maintained on disk. The Loader actually consists of three separate programs which provide assembly, debugging and IOC functions to the Emulator. It is a by-product of the Emulation, and it was used for implementation and testing of the Emulator. Its functions should be incorporated in the Emulator Program in the course of expansion to a full emulation.

```
                                $MERGE LOADER-SOURCE
                                PROGRAM UYK7-LOADER                                                                 00010C00 D
                                      %                                                                             00020000 D
                                ABASE=0                          %ADDRESS OF A-REG(0)                               0003000C D
                                BBASE=8                          %ADDRESS OF B-REG(0)                               00040C00 D
                                SBASE=16                         %ADDRESS OF S-REG(C)                               00050C00 D
                                PAR=29                           %ADDRESS OF P-REGISTER TEMP.                       0C060C00 D
                                SWITCH=88                        %LOCATIO( TO STORE SWITCH CHOICES                  0007CC00 D
                                PLOC=120                         %TEMPORARY ADDR. OF NEXT INSTRUCTION               0C080C00 D
                                HEXADDR=210                      %ADDRESS TO INSERT ADDRESSES TO PRINT              0C090C00 D
                                CARDBUF=212                      %PERMANENT ADDR. OF CARDBUFFER                     0C100C00 D
                                COL5X8=213                       %ADDRESS OF CARD COLS. K-8                         0C110C00 D
                                COL9X12=214                      %ADDRESS OF CARD COLS. 9-12                        0C120000 D
                                COL13X16=215                     %ADDRESS OF CARD COLS. 13-16                       00130C00 D
                                LASTCOL=232                      %ADDRESS OF COL 80 OF CARD                         0C140C00 D
                                PRINTADD=237                     %PERMANENT LOCATION OF PRINT BUFFER                0C150C00 D
                                      BLANKLINE=500              %LOACATION OF BUFFER OF SPACES                     0C160C00 D
                                      ERRORLIST=280. PRINTAREA=238. DISKADDR=3132                                   0017C000 D
                                      NUMLINES=36                %NUMBER OF LINES PRINTED X SIX                     00180C00 D
                                      %                                                                             0019CC00 D
                                      %BEGIN PROGRAM HERE BY ZEROING MEMORY                                         0020CC00 D
0C00  4800 0000 0030 00F0             WAIT                                                                          00210C00 D
                                START: %RETURN HERE ON SIGNAL FROM EMULATOR TO RESTART                              0C220C00 D
0C01  0D88 00C1 401C 00FC             SET GC1; WHEN GC1 THEN 1 L=A1,MAR2                                            0023CC00 D
0C02  0000 00C0 0030 0020             COMP 16 =: SAR                                                                0C240C00 D
0C03  4812 0000 0C30 00F0             0=: MIR; SAVE                                                                 00250C00 D
0C04  9809 A0DE 4030 1CF0             MW2: A1-1 =: A1; IF SAI                                                       00260C0C D
0C05  6819 0000 0030 00F0             IF ABT THEN SKIP ELSE STEP              %DONE WHEN A1 TOGGLES TO              00270C00 D
0C06  9C28 A0C3 0C1C 00F0             WHEN SAI THEN A1=: MAR2; JUMP                                                 0C280C00 D
                                      %                                                                             0029CC00 D
                                      %READ CARDS IN TO SETUP ERROR ROUTINES                                        0C3CCC00 D
                                ERRORSETUP:                                                                         0031CC00 D
0C07  4809 00D9 001C 00F0             B110=MAR2                                                                     0C320C00 D
0C08  4809 20C1 4030 00F0             LIT L=A1                                                                      0C330000 D
0C09  00C0 0000 0030 C0B0             MOST(DISKADDR)=LIT; COMP 8=SAR                                                00340000 D
0C0A  4809 A15C 4030 C0F0             A1 OR LIT=A1                                                                  0C350C00 D
0C0B  03CC 0003 0030 C0E0             LEAST(DISKADDR)=LIT                                                           0036CC00 D
0C0C  4809 2001 0B3C 00F0             LIT L=B                                                                       00370C00 D
0C0D  0B4C 00C3 0030 C0A0             COMP 16=SAR; 180=LIT          %READ FOUR SECTORS = 9 CARDS                    0C380C00 D
0C0E  4809 AC5C 0C30 00F0             A1 OR B=MIR                                                                   0C390C00 D
0C0F  0000 00C0 0030 006C             OUTPUT1-1=CPCR                                                                C0400C00 D
0C10  4809 0640 4030 C0F0             AMPCR=A1                                                                      00410000 D
0C11  118C 0000 0030 00C0             ERRORLIST=AMPCR                                                               00420C0C D
0C12  4809 2001 0B3C 00F0             LIT L=B                                                                       0C43CC00 D
0C13  0030 0000 CC00 00A0             3=LIT:COMP 16=SAR                                                             0C440C00 D
0014  4809 AC5C 0C90 C0F0             A1 OR B = MIR                                                                 00450C00 D
                                ERRORSETUP1:                                                                        0046CC00 D
0C15  0010 00C0 0030 0060             EXTOP-1=CPCR                                                                  0C470C00 D
0C16  014C 0C00 0030 0040             ERRORSETUP1-1=MPCR                                                            00480C00 D
                                      %                                                                             0C49CC00 D
                                      %SETUP BUFFER OF ALL SPACES FOR PRINTING BLANK LINES                          0C5CCC00 D
0C17  4809 0000 0901 00F0             0=BC8,LCTR                                                                    00510C00 D
0C18  0200 0003 0000 00E0             32=LIT                                                                        0C52CC0C D
0C19  1F40 00C0 0000 00CC             BLANKLINE=AMPCR               %ADDRESS OF BUFFER                              00530000 D
0C1A  4809 0C40 0030 00F0             B=MIR                                                                         00540C00 D
0C1B  4809 0643 001C 00F0             AMPCR=MAR2                                                                    00550C00 D
0C1C  4812 0000 0030 00F0             SAVE                                                                          0C560C00 D
0C1D  9809 0000 0000 1CF0             MW2:IF SAI                    %OUTPUT BLANKS = 30303C30                       0057CC00 D
```

```
0C1E  9C08 0F46 001E 00F0          WHEN SAI THEN BMAR+1=MAR2,INC                                    0058CC00 D
0C1F  8029 0000 0000 00F0          IF NOT COV THEN JUMP       %OUTPUT 33 BLANK WORDS                0059CC00 D
0C20  2809 0000 1030 C0F0          0=:A3;IF LC1             %INITIALIZE BASE REG TO 1C24            0060CC00 D
0C21  0020 0000 0030 C040          LOADERADDRESS=:MPCR        %ENTER AT LOADER ADDRESS+1 TO INITIL1006100C0 D
                                   %%CLEARS INPUT TO CARDS AND RETURNS HERE VIA NEWWRD              0062CC00 D
                                   %                                                                0063CC00 D
                                   %                                                                0064CC00 D
                                CHECKCOL1:                                                          0065CC00 D
0C22  4809 20C0 009C 00F0          LIT=:MAR2,MIR                          %BUFFER ADDRESS IS 198    0066CC00 D
0C23  0D40 0000 003C 00E0          CARDBUF=LIT                                                      0067CC00 D
0C24  003C 0000 0030 0060          INPUT-1=CPCR                                                     0068CC00 D
0C25  4809 2001 CC30 40F0          LIT C=A1,CSAR                                                    0069CC00 D
0C26  03F0 0000 0030 C090          63=:LIT;8=:SAR                                                   0070CC00 D
0C27  4809 AC56 8B3C 00F0          A1 AND B R =: B                        %CHAR IN B 0-7            0071CC00 D
0C28  4809 2C52 0000 00F0          LIT EOL B                  %CHECK F3 "D"                         0072CC00 D
0C29  0140 0000 0030 00E0          20=:LIT                                                          0073CC00 D
0C2A  6419 2C52 CC0C 00F0          IF FALSE THEN LIT EOL B;SKIP    %CHECK FOR " "                   0074CC00 D
0C2B  0040 0003 0000 0040          DATA-1=MPCR                                                      0075CC00 D
0C2C  030C 00C3 0C00 00E0          48=:LIT                    %SPACE                                0076CC00 D
0C2D  6419 2C52 0C00 C0F0          IF FALSE THEN LIT EOL B;SKIP   %CHE"K FOR "I"                    0077CC00 D
0C2E  0050 0000 0030 C04C          INSTRUCTION-1 =: MPCR                                            0078CC00 D
0C2F  019C 00C0 0000 00EC          25=:LIT                    %"I"                                  0079CCCC D
0C30  6419 2C52 0C00 C0F0          IF FALSE THEN LIT EOL B;SKIP %CHECK FOR "O"                      0080CC00 D
0C31  006C 0000 0030 0040          SETINDEXREG-1 =: MPCR                                            0081CC00 D
0C32  026C 0000 0030 C0E0          38=:LIT                    %"O"                                  0082CC00 D
0C33  6419 2C52 0000 C0F0          IF FALSE THEN LIT EOL B;SKIP %CHECK FOR "A"                      0083CC00 D
0C34  007C 0000 C030 0040          ORIGIN-1 =: MPCR                                                 0084CC00 D
0C35  0110 00C0 0C30 00E0          17=:LIT                    %"A"                                  0085CC00 D
0036  6419 2C52 0000 C0F0          IF FALSE THEN LIT EOL B;SKIP %CHECK FOR "S"                      0086CC00 D
0037  0080 0000 0000 0040          SETAREG-1 =: MPCR                                                00870000 D
0C38  032C 0000 0000 00E0          50=:LIT                    %"S"                                  0088CC00 D
0C39  6419 2C52 0C00 00F0          IF FALSE THEN LIT EOL B;SKIP  %CHEC5 FOR "N"                     0089CC00 D
0C3A  0090 0000 0C00 0040          SETSWITCHES-1 =: MPCR                                            0090CC00 D
0C3B  025C 0000 0030 C0E0          37=:LIT                    %"N"                                  0091CC00 D
0C3C  6419 2C52 0000 00F0          IF FALSE THEN LIT EOL B;SKIP  %CHEC5 FO "L"                      0092CC00 D
0C3D  00A0 0000 003C C040          IFETCH-1 =: MPCR                                                 0C93CC00 D
0C3E  023C 0000 0C30 C0E0          35=:LIT          %"L"=ADDRESS TO LOAD TO                         0094CC00 D
0C3F  6419 2C52 0C00 00F0          IF FALSE THEN LIT EOL B;SKIP  %CHEC5 FOR "O"                     0095CC00 D
0C40  00BC 0000 0030 0040          LOADERADDRESS-1=:MPCR                                            0096CC00 D
0041  0280 0000 0000 C0E0          40=:LIT                    %TEST FOR "O"                         0097CC00 D
0C42  6419 2C52 0000 C0F0          IF FALSE THEN LIT EOL B;SKIP  %CHECK FOR "R"                     0C98CC00 D
CC43  00C0 0000 0C30 0040          DUMP-1=:MPCR                                                     0C99CC00 D
0C44  029C 00C0 0000 00E0          41=LIT                     %"R" RESERVE AMOUNT OF SPACE IN 9-16  0100CC00 D
0C45  6419 2C52 0000 00F0          IF FALSE THEN LIT EOL B;SKIP  %CHCE"K FOR "W"                    0101CC00 D
0C46  00DC 0000 0000 0040          RESERVE-1=MPCR                                                   0102CCOC D
0C47  036C 0000 0000 00E0          54=LIT                     %"W" MEANS READ CHARACTER STRING       0103CC00 D
0C48  6419 0000 1000 00F0          IF FALSE THEN 0=A3;SKIP                                          0104CC00 D
0C49  00EC C000 0030 C040          WORD-1=MPCR                                                      0105CC00 D
004A  00F0 0000 0030 0040          PRINTERIO-1=MPCR                                                 0106CC00 D
                                   %                                                                0107CCOC D
                                   %                                                                0108CCOC D
                                   %                                                                01090C00 D
                                INSTRUCTION:                                                       01100000 D
                                   %                                                               01110C00 D
                                   %OPCODE IS IN CARD COLS. 3-4                                    01120CC00 D
                                   %ANALYZE OPCODE AND DETERMINE TYPE OF FORMAT.                   01130C00 D
                                   %    IF OPC <=60 => FORMAT1                                     01140000 D
                                   %    IF 62<=OPC<=67 THEN FORMAT 4B                              01150C00 D
                                   %   ELSE FORMAT 4A                                              01160000 D
0C4B  4809 0003 1018 00F0          0=:A3,BR2,LMAR             %FETCH WORC FROM 198                 0117CC00 D
```

```
0C4C  0D40 00C0 0000 C0E0          CARDBUF=LIT                                                    0118CCOO D
0C4D  0100 0000 0C30 0060          READAFIELD-1=:CPCR                                             01190COO D
0C4E  4809 2000 4000 C0F0          LIT=:A1                                                        0120CCOO D
0C4F  0070 00C0 0C30 00A0          7=LIT;COMP 16=SAR                                              0121CCOO D
0C50  4812 0C41 0B00 C0F0          B L=:B; SAVE                  %SETUP FOR TWICE THRU LOOP       0122OCOC D
                                   %LOOP BEGINS HERE                                              0123C000 D
                                   %                                                              0124OCOO D
0051  4809 E001 1030 C0F0          A3 L=:A3                      %MAKE ROOM FOR MSB               0125OCOO D
0052  9000 0003 0030 0030          COMP 3=:SAR                                                    0126CCOO D
0C53  4809 0C41 8B90 00FC          B C=:B,MIR                                                     0127CCOO D
0C54  0000 00C0 0000 0030          COMP 8=:SAR                                                    0128CCOC D
0C55  4809 AC56 0B00 00F0          A1 AND B=:B                                                    0129CCOO D
0C56  4836 EC5C 1D3C 00F0          B OR A3=:A3,BMI; CALL                                          0130CCOC D
                                   %END OF LOOP                                                   0131CCOO D
                                   %                                                              0132CCOO D
                                   %OPCODE IS NOW IN LSB OF B                                     01330CCO D
                                   %IF NOT AOV THEN OPC<60 =>TYPE 1                               01340CCO D
0057  4809 E15E 0000 00F0          A3 LSS LIT                                                     0135OCOO D
0C58  0300 0000 0030 00E0          48=:LIT                       %OPCODE = 60OCTAL                0136CCOO D
0C59  7C19 E158 0C00 00F0          IF FALSE THEN A3 LEQ LIT;SKIP                                  01370COO D
0C5A  0110 0000 0C30 C040          TYPE1-1=:MPCR                                                  0138CCOO D
                                   %IF NOT AOV THEN OPC<=61  =>TYPE 4A                            0139CCOO D
0C5B  0310 0000 0000 00E0          49=:LIT                       %OPCODE =61 OCTAL                0140CCOO D
0C5C  7C19 E15E 0000 00F0          IF FALSE THEN A3 GEQ LIT;SKIP                                  0141CCOO D
0C5D  0120 0000 0000 0040          TYPE4A-1=:MPCR                                                 0142CCOO D
                                   %IF AOV THEN OPCODE>=70  =>TYPE 4A                             0143CCOO D
                                   %ELSE TYPE 4D                                                  0144OCOC D
0C5E  038C 0000 0030 00E0          56=:LIT                       %OPC=7C OCTAL                    0145OCOO D
0C5F  7019 0000 0030 00F0          IF FALSE THEN SKIP                                             0146CCOO D
0C60  013C 0000 0000 0040          TYPE4A-1=:MPCR                                                 0147CCOO D
                                   %INSTRUCTION IS TYPE 4B - OPCODE STILL IN A3                   0148OCOO D
                              TYPE4B:                                                             0149OCOO D
0C61  0D50 00C0 0030 00E0          COL5X8=LIT                                                     0150OCOO D
0C62  014C 0000 0000 0060          READAFIELD-1=:CPCR            %FETCH COLS 5-8 INTO B           0151OCOO D
0C63  4809 E001 1000 C0F0          A3 L=:A3                                                       0152000C D
0C64  9000 00C0 0030 0030          COMP 3=:SAR                                                    0153OCOC D
0C65  4809 0C41 8B90 00F0          B C=:B,MIR                                                     0154OCOO D
0C66  0000 0000 0000 0030          COMP 8=:SAR                                                    0155CCOO D
0067  4809 AC56 0B00 00F0          A1 AND B=:B                   %A1 STILL HAS 7                  0156OCOO D
0C68  4809 EC5C 1D00 00F0          A3 OR B=:A3,BMI                                                0157CCOC D
0069  4809 0C41 8B90 C0F0          B C=:B,MIR                                                     0158CCOO D
0C6A  4809 E001 1030 C0FC          A3 L=:A3                      %MAKE ROOM FOR MSB OF "M"        0159OCOO D
0C6B  B00C 0000 0000 C030          COMP 1=:SAR                                                    0160CCOC D
0C6C  4812 E05C 1D00 00F0          A3 OR BOOT=A3,BMI;SAVE                                         0161OCOO D
                              %SETUP FOR TWO PASS LOOP FOR REST OF "M"                            0162CCOO D
0C6D  4809 E001 1030 00F0          A3 L=:A3                                                       0163OCOO D
0C6E  900C 00C0 0C30 0030          COMP 3=:SAR                   %GET NEXT DIGIT OF M TWICE       0164OCOO D
006F  4809 0C41 8B90 00F0          B C=:B,MIR                                                     0165CCOO D
0C70  0000 0000 0000 C030          COMP 8=:SAR                                                    0166OOOO D
0C71  4809 AC56 0B00 C0F0          A1 AND B=:B                                                    0167OCCO D
0C72  4836 EC5C 1D30 00F0          A3 OR B=:A3,BMI;CALL                                           0168OCOC D
                                   %END OF LOOP                                                   0169CCOO D
0C73  015C 0000 0030 0040          OUTPUT-1=:MPCR                %WORD WAS BUILT INTO LOWER A3    0170CCOO D
                                   %                                                              0171OCOC D
                              TYPE4A:          %OPCODE IS STILL IN A3                           0172CCOC D
0C74  0D5C 00C0 0030 00E0          COL5X8=LIT                                                     0173OCOO D
0C75  016C 0000 0000 0060          READAFIELD-1=:CPCR            %FETCH FIELDS A,F4,B,I           0174CCOO D
0C76  4852 0000 0000 00F0          SET LC2;SAVE                                                   0175CCOC D
                                   %START 3 PASS LOOP HERE                                        0176CCOO D
0C77  4809 E0C1 1000 00F0          A3 L=:A3                                                       0177CCOO D
```

```
0C78  9C00 0000 0030 0030          COMP 3=:SAR                                                  01780000 D
0C79  4809 0C41 8B80 00FC          B C=:B,MIR                                                   01790000 D
007A  0000 0C00 0000 0030          COMP 8=:SAR                                                  01800000 D
0C7B  4809 AC56 0B00 00F0          A1 AND B=:B                                                  01810000 D
0C7C  4809 EC5C 1D30 00F0          A3 OR B=:A3,BMI                                              01820000 D
0C7D  382E 00C0 0030 C0F0          IF LC2 THEN JUMP ELSE CALL                                   01830000 D
                                   %END LOOP                                                    01840000 D
                                   %                                                            01850000 D
0C7E  4809 E3C1 1C30 00FC          A3 L=:A3,BEX                                                 01860000 D
0C7F  B00C 00C0 0000 0030          COMP 1=:SAR                                                  01870000 D
                                   %INSERT I FIELD                                              01880000 D
0C80  4809 E05C 1030 00FC          A3 OR BOOT=:A3                                               01890000 D
0C81  0170 0000 0000 C040          OUTPUT-1=:MPCR              %DONE TYPE 4A                    01900000 D
                                   %RESULTS IN A3                                               01910000 D
                                   %                                                            01920000 D
                              TYPE1:          %OPCODE IS IN A3                                  01930000 D
0C82  0D50 0003 0030 00E0          COL5X8=LIT                                                   01940000 D
0C83  0180 0000 0000 0060          READAFIELD-1=:CPCR          %FETCH A,K,B,I FIELDS            01950000 D
0C84  4852 0000 0030 00F0          SET LC2;SAVE                                                 01960000 D
                                   %START 3 PASS LOOP HERE                                      01970000 D
0C85  4809 E001 1C30 00F0          A3 L=:A3                                                     01980000 D
0C86  9000 00C0 0030 0030          COMP 3=:SAR                                                  01990000 D
0C87  4809 0C41 8B80 00F0          B C=:B,MIR                                                   02000000 D
0C88  000C 0003 0030 C03C          COMP 8=:SAR                                                  02010000 D
0C89  4809 AC56 0B0C C0F0          A1 AND B=:B                                                  02020000 D
0C8A  4809 EC5C 1D3C 00F0          A3 OR B=:A3,BMI                                              02030000 D
0C8B  382E 00C0 0000 00F0          IF LC2 THEN JUMP ELSE CALL                                   02040000 D
                                   %END LOOP A,K,B FIELDS INSERTED NOW GET I FIELD              02050000 D
0C8C  4809 E001 1C30 00F0          A3 L=:A3,BEX                                                 02060000 D
0C8D  B00C 0000 0000 C030          COMP 1=:SAR                                                  02070000 D
0C8E  4809 E05C 1030 00F0          A3 OR BOOT=:A3                                               02080000 D
0C8F  0D60 00C0 0030 00EC          COL9X12=LIT                                                  02090000 D
0090  0190 0000 0C30 0060          READAFIELD-1=:CPCR                                           02100000 D
0091  4809 E0C1 1C00 C0F0          A3 L=:A3                                                     02110000 D
0092  9000 0000 0000 003C          COMP 3=:SAR                 %INSERT S FIELD                  02120000 D
0093  4809 0C41 8B80 00F0          B C=:B,MIR                                                   02130000 D
0094  000C 0000 0C30 C030          COMP 8=:SAR                                                  02140000 D
0095  4809 AC56 0B30 00F0          A1 AND B=:B                                                  02150000 D
0096  4809 EC5D 1D3C 00F0          A3 OR B L=:A3,BMI                                            02160000 D
0097  B000 0000 0030 C030          COMP 1=:SAR                                                  02170000 D
0098  4809 0C41 8B8C 00F0          B C=:B,MIR                                                   02180000 D
0099  0000 0000 0C30 0030          COMP 8=:SAR                                                  02190000 D
                                   %MSB OF Y HAS BEEN INSERTED                                  02200000 D
009A  4849 E05C 1D00 00FC          A3 OR BOOT=:A3,BMI;SET LC2                                   02210000 D
                                   %START TWO - TWO PASS LOOPS HERE FOR REMAINDER OF Y FIELD    02220000 D
009B  4812 0000 0030 C0F0     YFIELD: SAVE                                                      02230000 D
009C  4809 E0C1 1030 C0F0          A3 L=:A3                                                     02240000 D
009D  9000 0003 0000 C030          COMP 3=:SAR                                                  02250000 D
009E  4809 0C41 8B80 00F0          B C=:B,MIR                                                   02260000 D
009F  0000 00C0 0030 0030          COMP 8=:SAR                                                  02270000 D
00A0  4809 AC56 0B00 00F0          A1 AND B=:B                                                  02280000 D
00A1  4836 EC5C 1D30 00F0          A3 OR B=:A3,BMI; CALL                                        02290000 D
                                   %END OF LOOP                                                 02300000 D
                                   %                                                            02310000 D
00A2  0D7C 00C3 0000 00E0          COL13X16=LIT                                                 02320000 D
00A3  01A0 0000 003C C06C          READAFIELD-1=:CPCR                                           02330000 D
00A4  3419 E0C0 0030 00F0          IF NOT LC2 THEN A3=MIR; SKIP ELSE STEP                       02340000 D
00A5  09AC 0000 0000 0040          YFIELD-1=:MPCR                                               02350000 D
                                   %COMPLETED Y                                                 02360000 D
                                   %COMPLETED TYPE 1 NOW SAVE IN MIR AND OUTPUT                 02370000 D
```

```
0CA6  01B0 0003 0000 0040          NOPACK-1=MPCR                                          02380000 D
                                   %                                                      02390000 D
                                   %IF TYPE 4 INSTRUCTION THEN IF LOWER HALF OF PREVIOUS  02400000 D
                                   %INSTRUCTION WAS ZERO AND IN THIS INSTRUCTION AND      02410000 D
                                   %AND OUTPUT, ELSE SHIFT THIS INSTRUCTION TO UPPER &ALF 02420000 D
                                   %AND OUTPUT AND ADVANCE PAR.                           02430000 D
                                   %                                                      02440000 D
                                OUTPUT:                                                   02450000 D
0CA7  4809 C00E 001C 00FC          A2-1=:MAR2                                             02460000 D
0CA8  A809 C000 0030 0CF0          MR2; IF RDC                %READ PREVIOUS INSTRUCTION  02470000 D
0CA9  AC08 0000 0C30 00F0          WHEN RDC THEN BEX                                      02480000 D
0CAA  4809 0C41 0B30 00F0          B L=:B                     %CHECK LOWER HALF FOR ZEROS 02490000 D
0CAB  000C 0000 0030 0020          COMP 16=:SAR                                           02500000 D
0CAC  01CC 0000 0000 00C0          PACKLEFT-1=:AMPCR          %IF NOT THEN WRITE UPPER ONLY 02510000 D
0CAD  4809 0C42 0000 C0F0          NOT B                                                  02520000 D
0CAE  6C0D 0000 0C30 00F0          IF ABT THEN BEX;STEP ELSE JUMP                         02530000 D
0CAF  4809 EC5C 0030 00F0          A3 OR B=:MIR               %JOIN INSTRUCTION INTO LOWER &ALF 02540000 D
                                WRITEOUT:                                                 02550000 D
0CB0  01DC 0000 0000 0040          OUT-1=MPCR                                             02560000 D
                                PACKLEFT:                                                 02570000 D
0CB1  4809 E001 0030 00F0          A3 L=:MIR                  %MOVE INSTRUCTION TO UPPER HALF 02580000 D
0CB2  0000 000C 0030 0020          COMP 16=:SAR                                           02590000 D
0CB3  4809 C00D 001C 00F0       NOPACK: A2=MAR2                                           02600000 D
0CB4  4809 C0C0 2C00 00F0          A2+1=:A2                   %UPDATE PAR                 02610000 D
0CB5  01EC 0000 0030 0040          OUT-1=MPCR                                             02620000 D
                                   %                                                      02630000 D
                                READAFIELD:    %RETURN CONTENTS OF BUFFER ADDRESSES BY LIT 02640000 D
0CB6  4809 2003 001C C0F0          LIT=:MAR2                                              02650000 D
0CB7  A809 0000 0000 0CF0          MR2; IF RDC                                            02660000 D
0CB8  AC28 0000 0C00 00F0          WHEN RDC THEN BEX; JUMP                                02670000 D
                                   %END OF INSTRUCTION SETUP ROUTINES                     02680000 D
                                   %                                                      02690000 D
                                %                                                         02700000 D
                                ORIGIN:                                                   02710000 D
0CB9  01F0 00C0 0000 0060          GETADDR-1=:CPCR                                        02720000 D
                                   %GET NEW ADDRESS AND PUT INTO P-REG LOCATION COUNTER 106 02730000 D
0CBA  4809 2003 001C 00F0          LIT=:MAR2                                              02740000 D
0CBB  078C 0000 0000 00A0          PLOC=LIT;16=SAR                                        02750000 D
0CBC  4809 00C1 0B30 00F0          1 L=:B                     %OFFSET BY 1024             02760000 D
0CBD  A000 0000 000C 0020          COMP 10=:SAR                                           02770000 D
0CBE  4809 EC40 2030 00F0          A3+B=MIR,A2                                            02780000 D
0CBF  0200 0000 0C30 C040          OUT-1=MPCR                                             02790000 D
                                   %                                                      02800000 D
                                   %                                                      02810000 D
                                SETINDEXREG:    %GET REG NUMBER FROM WORD IN COL 7-8, BUFFER 199 02820000 D
                                   %SAVE IN A3.AFTER OFFSETTING BY 9. PASS CONTROL TO SETREG 02830000 D
                                   %WHICH WILL DECIPHER VALUE AND STORE IN PROPER REGISTER. 02840000 D
                                   %MAR ASSUMED STILL SET AT 198                          02850000 D
                                   %                                                      02860000 D
00C0  0210 0000 0000 0060          GETADDR-1=CPCR              %VALUE RETURNED IN A3      02870000 D
00C1  4809 E140 1030 00F0          A3 + LIT =:A3                                          02880000 D
00C2  0080 0000 0C00 00E0          BBASE=LIT                                              02890000 D
00C3  0220 00C0 0030 C04C          SETADDR-1=:MPCR                                        02900000 D
                                   %                                                      02910000 D
                                   %                                                      02920000 D
                                SETAREG:          %GET REG NUMBER FROM COL 7-8 WORD 199 SAVE IN A3. 02930000 D
                                   %PASS CONTROL TO SETREG WHICH WILL DECIPHER VALUE AND  02940000 D
                                   %STORE IN REG.MAR STILL SET TO REREAD 198              02950000 D
                                %                                                         02960000 D
00C4  0230 00C0 0030 006C          GETADDR-1 =:CPCR            %VALUE RETURNS IN A3       02970000 D
```

```
00C5  0240 0000 0030 C040          SETADDR-1=:MPCR                                          02980C00 D
                                   %END SETAREG                                             02990000 D
                                   %                                                        03000C00 D
                                SETSWITCHES:                                                03010C00 D
                                   %%SWITCH VALUE WILL BE IN COL 4 OF CARD OR BITS 12-14 OF A3   03020C0C D
                                   %MASK VALUE AND WRITE TO ADDRESS 64                      03030C0C D
00C6  025C 0000 0030 0060          GETADDR-1=:CPCR                                          03040C00 D
00C7  4809 E0C0 0030 00FC          A3=MIR                                                   03050C00 D
00C8  0580 00C0 0C00 00E0          SWITCH=LIT                                               0306CC00 D
00C9  4809 20C3 001C 00F0          LIT=MAR2                                                 03070C00 D
00CA  026C 0000 0030 0040          OUT-1=MPCR                                               0308CC0C D
                                   %                                                        03090C00 D
                                   %                                                        03100C00 D
                                IFETCH:                                                     0311CC00 D
                                   %ROUTINE SETS UP PAR AND GOES TO A LOOP CHECKING GC1     03120C0C D
                                   %WHICH WILL BE SIGNAL TO DO I/O. RETURNS TO OTHER PROCESSOR   0313CC0C D
                                   %AFTER CLEARING GC1                                      03140C00 D
00CB  0270 0000 0030 0060          GETADDR-1=:CPCR                                          03150C0C D
00CC  4809 E0C0 0030 C0F0          A3=MIR                                                   03160C00 D
00CD  4809 20C3 001C 00F0          LIT=:MAR2                                                0317CC00 D
00CE  01D0 00C0 CC30 00E0          PAR=LIT                                                  0318CC00 D
00CF  028C 00C0 0C00 C060          OUTPUT1-1=CPCR                                           03190C0C D
00D0  4949 00C0 0000 C0FC          RESET GC1                                                03200C00 D
00D1  0290 00C0 0000 0040          IOREQ-1=MPCR                                             03210C00 D
                                   %                                                        03220000 D
                                   %                                                        03230C00 D
                                LOADERADDRESS:  %SETUP ADDRESS TO LOAD PROGRAM INTO. ADDRESS WILL BE   0324CC0C D
                                                %ADDED TO 1024 FOR CORRECT LOADING. NUMBER WILL BE WRITT0325CC00 D
                                   %8 BASE REGISTERS      LOCATION COUNTER WILL BE STORED AT 20C   03260000 D
                                   %AND CAN ALSO BE MODIFIED BY "D" CARD                    0327CC00 D
                                   %SET BASE REGISTER IN INCREMENTS OF 8192                 03280C00 D
                                   %                                                        03290C00 D
00D2  02A0 0000 0030 0060          GETADDR-1=:CPCR            %LOCATION RETURND IN A3       03300C00 D
00D3  4809 00C1 0B30 C0F0          1 L=B,MIR                                                03310C00 D
00D4  A100 00C0 0030 C0A0          COMP 10=SAR;SBASE=LIT                                    03320C00 D
00D5  4809 EC40 1030 00F0          A3+B=A3                    %ADD 1024 TO OFFSET ADDRESS   03330C0C D
00D6  4809 20C3 001C 00F0          LIT=MAR2                                                 03340C00 D
00D7  4809 00C1 4C00 00FC          1 L=A1                                                   0335CC00 D
00D8  3070 00C3 0030 C0A0          7=LIT;COMP 13=SAR                                        03360C00 D
00D9  4812 0000 0001 C0F0          LCTR;SAVE                                                03370C00 D
00DA  9809 000B 0030 1CF0          MW2;BMI;IF SAI                                           03380C00 D
00DB  9C08 AC40 0C32 C0F0          WHEN SAI THEN A1+B=MIR,INC                               03390C00 D
00DC  8429 0F45 001C 00F0          IF NOT COV THEN BMAR+1=MAR2;JUMP                         03400C00 D
00DD  4809 E00D 0B30 00F0          A3=MIR,B                                                 0341CC00 D
00DE  4809 20C3 001C 00F0          LIT=MAR2                   %ADDRESS OF LOCATION COUNTER  03420C00 D
00DF  078C 00C3 0030 00E0          PLOC=LIT                                                 03430C00 D
                                   %A2 NOW HAS SECTOR ADDRESS/LOCATION COUNT                03440C00 D
00E0  4809 C0C0 A030 40F0          A2 R=:A2,CSAR                                            0345CC0C D
00E1  0000 0000 0030 0020          16=:SAR                    %GET LOAD ADDRESS INTO LOW ORDER OF AC3460C00 D
00E2  4809 CC01 200C C0F0          A2 L=:A2                   %A2=SECTOR ADDRESS/LOCATION COUNTER  0347CC00 D
00E3  4809 CC5C 2030 00F0          A2 OR B =:A2                                             03480C0C D
00E4  9809 000B 0000 1CFC      OUT:MW2;IF SAI                                               03490000 D
00E5  9C08 0000 0030 00F0          WHEN SAI THEN 0; STEP                                    0350CC00 D
                                   %8 REGISTERS 24-31 SET AND LOCATION COUNTER 106 SET      0351C000 D
                                   %                                                        03520C00 D
                                %                                                           03530000 D
                                NEWWRD:                                                     0354CC00 D
                                CARDREAD:                                                   03550000 D
                                   %ROUTINE USES A1,B,MIR                                   0356CC0C D
                                   %THIS ROUTINE PLACES A 0/198 INTO MAILBOX                0357CCC0 D
```

```
                                %THIS SHOULD READ 20 WORDS INTO ADDRESS 334(8)                                  03580C00 0
0CE6  4809 C000 C931 C0F0       0=BC8,LCTR                                                                      03590C00 0
0CE7  0210 00C0 0030 00E0       33=LIT                                                                          03600C00 0
0CE8  4809 20C3 001C 00F0       LIT=MAR2                                                                        03610C00 0
0CE9  002C 00C0 003C C0A0       HEXADDR=LIT;COMP 16=SAR                                                         03620C00 0
0CEA  4809 0C40 0030 C0F0       B=MIR                                                                           03630C00 0
0CEB  0280 0C00 0000 C060   ZBUFF:  OUTPUT1-1=CPCR                                                              03640C00 0
0CEC  840B 0F46 0C1E C0F0       IF NOT COV THEN BMAR+1=MAR2,INC;STEP ELSE SKIP                                  03650C00 0
0CED  0EA0 00C3 0C30 0040       ZBUFF-1=MPCR                                                                    03660C00 0
0CEE  9809 2000 0030 00F0   CARDFETCH:          LIT =: MIR; IF SAI                                              03670C00 0
0CEF  0040 00C0 0030 00E0       CARDBUF=LIT                                                                     03680C00 0
0CF0  02CC 00C0 C030 C060       EXTOP-1 =: CPCR                                                                 03690C00 0
0CF1  0E0C C000 003C 0040       CAROFETCH-1 =: MPCR         %COULDNOT READ CARD                                 03700C00 0
0CF2  4809 C001 4C30 00F0       A2 L=A1                     %PUT ADDRESS IN A1                                  03710C00 0
0CF3  8D2C 00C3 0030 C090       HEXADDR=LIT;COMP 20=SAR                                                         03720C00 0
0CF4  4809 20C3 001C 00FC       LIT=MAR2                    %ADDRESS BEFORE CAROBUFFER                          03730C00 0
0CF5  4812 0000 C031 C0F0       0=MIR,LCTR;SAVE             %LOOP 4 TIMES FOR ADDRESS                           03740C00 0
0CF6  003C 0000 0000 00B0       COMP 8=SAR;3=LIT                                                                03750C00 0
0CF7  9809 0C41 0032 C0F0       B L=MIR,INC;IF SAI                                                              03760C00 0
0CF8  4809 A0C0 8F3C C0F0       A1 R=BBI                                                                        03770C00 0
0CF9  9000 0000 0030 0030       COMP 3=SAR                                                                      03780C00 0
0CFA  4809 A0C1 4C30 00FC       A1 L=A1                     %MSB OR 7IR                                         03790C00 0
0CFB  4809 0C40 0030 C0F0       B=MIR                                                                           03800C00 0
0CFC  8A0D 0000 0C00 1CF0       IF COV THEN MW2;STEP ELSE JUMP                                                  03810C00 0
0CFD  9C08 2000 003C C0F0       WHEN SAI THEN LIT=MIR,MAR2                                                      03820C00 0
0CFE  0D40 00C0 003C C0BC       CAROBUF=LIT;24=SAR                                                              03830C00 0
0CFF  0200 00C0 0030 0060       INPUT-1=CPCR                %CHECK F3R INSTRUCTION                              03840C00 0
0100  4809 0C40 8B30 C0FC       B R=B                       %  OR CONTROL CARD                                  03850C00 0
0101  4809 2C52 0000 C0FC       LIT EQL B                                                                       03860C00 0
0102  0300 00C0 0000 00A0       48=LIT;16=SAR               %SPACE " "                                          03870C00 0
0103  6C00 2000 0C3C C0FC       IF TRUE THEN LIT=MIR;STEP ELSE SKIP                                             03880C00 0
0104  0020 00C0 0030 00A0       HEXADOR=LIT;16=SAR                                                              03890C00 0
0105  BC19 00C1 0F00 00F0       IF IRO THEN 1 L=BBI;SKIP                                                        03900C00 0
0106  021C 00C0 0030 C040       CHECKCOL1-1=MPCR                                                                03910C00 0
0107  4809 0C40 003C 00F0       B=MIR                                                                           03920C00 0
                            CARDPRINT:                      %ECHO CARD READ                                     03930C00 0
0108  02EC 00C0 003C C060       EXTOP-1 =: CPCR                                                                 03940C00 0
0109  1070 00C0 0C30 0040       CARDPRINT-1=MPCR            %ERROR RETURN FOR PRINTING                          03950C00 0
010A  0210 0000 0C30 C040       CHECKCOL1-1=MPCR                                                                03960C00 0
                                %                                                                               03970C00 0
                                %                                                                               03980C00 0
                            EXTOP:              %THIS ROUTINE PERFORMS REQUESTED EXTERNAL INTERFA03990C00 0
                                                %TO THE IOP MIR = FUNCTION/ADDRESS                              04000C00 0
                                                %RETURNS CONDITION OF OP IN B                                   04010C00 0
010B  1C88 0002 0C2C C0F0       SET GC2 WHEN GC2 THEN NOT 0 =: MAR1                                             04020000 0
010C  9809 00C0 0C30 18F0       MW1; IF SAI                                                                     04030C00 0
010D  9F08 0000 0C30 00F0       WHEN SAI THEN 0; SET INT                                                        04040C00 0
010E  0809 00C0 0000 C0F0       IF INT                                                                          04050C00 0
010F  0808 0000 0030 00F0       WHEN INT THEN STEP                                                              04060C00 0
0110  A809 0003 0C30 C8F0       MR1;IF ROC                                                                      04070C00 0
0111  AEC8 0C00 0C3C C0F0       WHEN RDC THEN BEX; RESET GC2                                                    04080C00 0
0112  4809 0C42 0C0C 00F0       NOT B                       %B=0 IS ERROR RETURN ELSE JUMP                      04090C00 0
0113  682F C000 0000 C0F0       IF ABT THEN JUMP ELSE RETN                                                      04100C00 0
                                %END EXTERNAL FUNCTION                                                          04110C00 0
                                %                                                                               04120C00 0
                            GETADOR:                                                                            04130C00 0
                                %ASSUME MAR2 IS SET UP TO REREAD SAME WORD VIA BEX. RESULT WILL 04140C00 0
                                %BE RETURNED IN A3, 7MIR CONTAINS ADDRESS OF WORD                               04150C00 0
0114  4809 C0C0 0C00 00F0       BEX                                                                             04160C00 0
0115  4809 0003 1011 00FC       0 =: A3,BR2,LCTR            %FIRST DIGIT IS LAST BYTE OF B                      04170C00 0
```

```
0116  0030 0000 0000 00B0          COMP 8 =:SAR;3=:LIT                                          04180000 D
0117  4809 2057 1000 00F0          B AND LIT L =: A3          %COL 4 NOW IN A3 (4-6)            04190000 D
0118  9070 0000 0000 00B0          7=:LIT;COMP 3=;SAR                                           04200000 D
0119  4809 2003 001C 00F0          LIT=:MAR2                                                    04210000 D
011A  0D50 0000 0000 00E0          COL5X8=LIT                                                   04220000 D
011B  A809 0000 0000 00F0          MR2;IF RDC                                                   04230000 D
011C  AC08 0000 0000 00F0          WHEN RDC THEN BEX                                            04240000 D
011D  4809 0041 8890 00F0  GETADDRLOOP:    B C =: B,MIR       %SHIFT TO GET BYTE LSB AND SAVE   04250000 D
011E  0070 0000 0000 00B0          COMP 8 =: SAR; 7 =: LIT                                      04260000 D
011F  4809 2056 0B02 00F0          B AND LIT =: B,INC                                           04270000 D
0120  4809 E05D 1D10 40F0          A3 OR B L=A3,BMI,CSAR                                        04280000 D
0121  9000 0000 0000 0030          COMP 3 =: SAR                                                04290000 D
0122  8029 E000 9000 00F0          IF COV THEN A3 R=A3; JUMP ELSE STEP                          04300000 D
0123  1100 0000 0000 0040          GETADDRLOOP-1 =: MPCR                                        04310000 D
                                   %                                                            04320000 D
                                   %                                                            04330000 D
                           SETADDR:        %SET THE REGISTER ADDRESSED IN A3 TO VALUES IN COLS 9-16 04340000 D
                                           %FOUND IN BUFFERS 199 & 200                          04350000 D
                                   %                                                            04360000 D
0124  4849 2003 001C 00F0          LIT=MAR2;SET LC2                                             04370000 D
0125  0D60 0000 0000 00E0          COL9X12=LIT                                                  04380000 D
                                   %GET COLUMNS 9-12                                            04390000 D
0126  02F0 0000 0000 0060          INPUT-1=CPCR                                                 04400000 D
0127  4809 0040 4000 00F0          B=A1                       %A1 HAS VALUE                     04410000 D
0128  4809 A001 4001 00F0          A1 L=A1,LCTR               %GET 2 MSB LEFT                   04420000 D
0129  2020 0000 0000 00B0          COMP 6=SAR;2=LIT                                             04430000 D
012A  4809 A000 8B00 00F0          A1 R=B                     %B HAS 2MSB OF NEXT 3 CHARS       04440000 D
012B  A000 0000 0000 003C          COMP 2=SAR                                                   04450000 D
012C  4812 A001 4000 00F0          A1 L=A1;SAVE                                                 04460000 D
012D  4809 A001 4000 00F0  SETLOOP:  A1 L=A1                  %GET NEXT 3 BITS LEFT             04470000 D
012E  3000 0003 0000 0030          COMP 5=SAR                                                   04480000 D
012F  4809 0041 0000 00F0          B L=MIR                                                      04490000 D
0130  9000 0000 0000 0030          COMP 3=SAR                                                   04500000 D
0131  4809 A000 8F00 00F0          A1 R=BBI                                                     04510000 D
0132  4809 A001 4002 00F0          A1 L=A1,INC                                                  04520000 D
0133  4809 0040 0000 00F0          B=MIR                                                        04530000 D
0134  8019 0F46 0D1C 00F0          IF COV THEN BMAR+1=MAR2,BMI;SKIP                             04540000 D
0135  1200 0000 0000 0040          SETLOOP-1=MPCR                                               04550000 D
0136  A809 0000 0000 00F0          MR2;IF RDC. WHEN RDC THEN BEX                                04560000 D
0137  AC08 0000 0000 00F0
0138  4809 0040 4D01 00F0          B=A1,LCTR,BMI                                                04570000 D
0139  0030 0000 0000 00E0          3=LIT                      %RETURN 3 TIMES FOR COLS 17-19    04580000 D
013A  3435 0000 0002 00F0          IF NOT LC2 THEN INC; CALL ELSE JUMP                          04590000 D
013B  4809 E003 001C 00F0          A3=MAR2                                                      04600000 D
013C  0E50 0003 0000 00C0          NEWWRD-1=AMPCR                                               04610000 D
                           OUTPUT1:                                                             04620000 D
013D  9809 0003 0000 1CF0          MW2;IF SAI                                                   04630000 D
013E  9C28 0000 0000 00F0          WHEN SAI THEN 0;JUMP                                         04640000 D
013F  A809 0000 0000 00F0  INPUT:  MR2;IF RDC                                                   04650000 D
0140  AC28 0000 0000 00F0          WHEN RDC THEN BEX;JUMP                                       04660000 D
                                   %                                                            04670000 D
                           DUMP:                                                                04680000 D
0141  1130 0000 0000 0060          GETADDR-1=:CPCR            %A3=ADDRESS IN COLS.4-8           04690000 D
                           IODUMP:                                                              04700000 D
0142  4809 C000 0090 00F0          A2=;MIR                                                      04710000 D
                                   %A2 SAVED AWAY AS TEMPORARY FOR FUTURE RESTORATION           04720000 D
0143  4809 2003 001C 00F0          LIT=MAR2                                                     04730000 D
0144  0780 0003 0000 00A0          PLOC=LIT;COMP 16=SAR                                         04740000 D
0145  4809 2000 0B00 00F0          LIT=B                                                        04750000 D
0146  0240 0000 0000 00E0          NUMLINES=LIT               %NUMBER OF ADDRESSES TO PRINT     04760000 D
```

```
0147  13C0 C000 0C00 C06C          OUTPUT1-1=:CPCR                                          04770C00 D
0148  4809 2001 2000 00F0          LIT L =:A2                                               04780000 D
0149  0060 00C0 0C30 C0EC          6=:LIT                          %WORDS PER LINE=6        04790C0C D
014A  4809 CC5C 2030 C0F0          A2 OR B=:A2                     %A2=6/65=WDS PER LINE/65 WORDS 04800C00 D
014B  4809 E0C1 1000 00F0    SETBUFF: A3 L=A3                                               04810C00 D
014C  0EDC 00C0 0C00 C0A0          COMP 16=SAR;PRINTAREA-1=LIT                              04820000 D
014D  4809 E15C 1C00 00F0          A3 OR LIT=:A3                   %A3=READ ADDR/BUFFER ADDR 04830C00 D
014E  4809 0CC0 0930 C0F0          0=:BC8                          %B=30303030              04840C00 D
014F  4809 0C40 0081 C0F0          B=MIR,LCTR                                               04850C00 D
0150  021C 0000 0C00 C0E0          33=:LIT                                                  04860C00 D
0151  4809 2003 001C C0FC          LIT=MAR2                                                 04870C00 D
0152  0EEC 0000 0C00 00EC          PRINTAREA=LIT                                            04880C00 D
                             ZEROLOOP:                                                     04890C00 D
0153  13C0 0C00 0030 C060          OUTPUT1-1=:CPCR                                          049C0C00 D
0154  840B 0F46 001E C0F0          IF NOT COV THEN BMAR+1=MAR2,INC; STEP ELSE SKIP          04910000 D
0155  152C 0000 0C30 C04C          ZEROLOOP-1=:MPCR                                         04920C00 D
0156  4809 E0C0 C030 C0FC          A3 R=:A1                        %START BY PRINTING ADDRESS 0493C000 D
0157  0000 0000 0C3C 0020          16=:SAR                                                  04940C00 D
0158  4809 0000 0901 C0FC    CONVERT:  0=:BC8,LCTR                                          0495C000 D
0159  2010 0000 0030 C0BC          1=LIT;COMP 6=SAR                                         04960C00 D
015A  4809 0C41 0B30 C0F0          B L=:B                                                   04970C00 D
015B  4809 0C41 0080 00F0          B L=:MIR                                                 0498C000 D
015C  A00C 00C0 0C00 C030          COMP 2=SAR                                               04990C00 D
015D  4809 A0C0 8F30 C0F0          A1 R=:BBI                                                05000C00 D
015E  4809 A001 4030 00F0          A1 L=:A1                                                 05010C00 D
015F  4852 0003 0000 C0F0          SET LC2;SAVE                                             0502C000 D
                             CONVLOOP:                                                     05030C00 D
0160  4809 0C41 0B3C C0F0          B L=:B                          %BEGIN LOOP HERE         05040C00 D
0161  3000 00C0 0030 C03C          COMP 5=:SAR                                              05050C0C D
0162  4809 0C41 0C80 0CF0          B L =:MIR                       %SHIFT A1 INTO B         0506CC00 D
0163  9000 00C0 0030 C030          COMP 3=:SAR                                              05070C00 D
0164  4809 A0C0 8F30 00F0          A1 R=:BBI                                                05080C00 D
0165  4809 A001 4002 00F0          A1 L=:A1,INC                                             05090C00 D
0166  8C19 E0C3 001C C0F0          IF COV THEN A3=MAR2;SKIP ELSE STEP                       050CCC00 D
0167  15FC 0000 0C30 0040          CONVLOOP-1=:MPCR                                         0511CC0C D
0168  4809 E0C0 100C 00F0          A3+1=:A3                                                 05120000 D
0169  4809 0C40 0090 00F0          B=:MIR                          %B CONTAINS CONVERTED WORD 05130C00 D
016A  9809 0000 0030 1CFC          MW2; IF SAI                                              05140CC0 D
016B  9C08 00C0 0901 C0F0          WHEN SAI THEN 0=:BC8,LCTR                                0515C000 D
016C  003C 0000 0000 00E0          3=:LIT                                                   05160C00 D
016D  382E 00C0 0000 00FC          IF LC2 THEN JUMP ELSE CALL                               05170C00 D
                                   %END OF LOOP ONE 32 BIT WORD WRITTEN IN 3 WORDS          05180C0C D
016E  4809 C0C1 0B30 00F0          A2 L=:B                         %B=(UMBER WORDS/0        0519CC00 D
016F  0060 00C0 0030 00AC          COMP 16=:SAR;6=:LIT                                      0520CCC0 D
0170  0300 0000 00BC 00C0          FETCHWORD-1=:AMPCR                                       0521CC0C D
0171  4809 C0C0 A030 C0FC          A2 R=:A2                                                 05220C00 D
0172  4809 C0DE 203C C0F0          A2-1=:A2                        %SEE IF DONE ONE LINE YET 0523CC00 D
0173  4429 CC5D AC30 C0F0          IF NOT MST THEN A2 OR B C=:A2;JUMP                       0524CC00 D
0174  4809 2C41 AC30 C0FC          LIT + B C=:A2                                            05250C00 D
0175  031C 0000 0000 0040          PRINTBUFF-1=:MPCR                                        0526CC0C D
0176  4809 E0C0 1C30 C0F0    FETCHWORD:  A3+1=A3                                            05270C00 D
0177  4809 E0C1 901C 00F0          A3+1 C=A3,MAR2                                           05280C00 D
0178  13E0 0000 0000 0060          INPUT-1=:CPCR                                            05290C00 D
0179  4809 C0DE 2005 C0F0          A2-1=:A2,CTR                                             05300C00 D
017A  4809 0C40 4002 C0FC          B=A1,INC                                                 0531CC0C D
017B  840B E0C1 9C00 C0F0          IF NOT COV THEN A3+1 C=A3;STEP ELSE SKIP                 0532CC00 D
017C  157C 0000 003C C040          CONVERT-1=:MPCR                                          05330C00 D
017D  4809 2003 001C C0FC          LIT=:MAR2                       %DONE ALL 65 WORDS       05340C00 D
017E  0780 0003 0000 00E0          PLOC=LIT                                                 05350C00 D
017F  13EC 0000 003C 0060          INPUT-1=:CPCR                                            05360C00 D
```

```
0180  4809 0C40 2000 C0F0          B=:A2                          %RESTORE A2 FROM SAVED LOCATION      05370C00 D
0181  0E50 00C0 0000 C0C0          NEWWRD-1=AMPCR                                                      05380C00 D
0182  2029 0000 0C00 00FC          IF NOT LC1 THEN JUMP                                                05390C00 D
0183  0320 0000 0000 C040          IORETURN-1=MPCR                                                     05400C00 D
                                   %                                                                   05410C00 D
                              PRINTBUFF:                                                               05420C00 D
0184  4809 00C1 0B00 C0F0          1 L=:B                                                              05430C00 D
0185  0EE0 00C0 0000 00AC          COMP 16=SAR;PRINTAREA=LIT                                           05440C00 D
0186  4809 2C5C 0C9C C0F0          LIT OR B=:MIR                  %B=1/ADDRESS FOR PRINT BUFFER CODE   05450C00 D
0187  10A0 0000 0C00 006C          EXTOP-1=:CPCR                                                       05460C00 D
0188  1830 00C0 0000 C040          PRINTBUFF-1=MPCR                                                    05470C00 D
0189  4809 E0C0 9C00 C0F0          A3 R=A3                                                             05480C00 D
018A  14A0 0000 000C C040          SETBUFF-1=:MPCR                                                     05490C00 D
                              DATA:                               %INPUT DECIMAL DATA                05500C00 D
018B  49C9 20C3 00EC C0FC          LIT=MAR2; SET LC1                                                   05510C00 D
018C  0D60 0000 0C90 00E0          COL9X12=LIT                                                         05520C00 D
018D  4809 00C0 0C91 00FC          0=MIR,LCTR                                                          05530C00 D
018E  0030 0000 0C00 C0EC          3=LIT                                                               05540C00 D
018F  13E0 0000 0000 006C     DATA1: INPUT-1=CPCR                                                      05550C00 D
0190  4812 0C40 4D00 C0F0          B=A1,BMI;SAVE                                                       05560C00 D
0191  4809 A0C1 40CC C0FC          A1 L=A1                        %GET RID OF BCL HEADER               05570C00 D
0192  8000 0000 0C00 0030          COMP 4=SAR                                                          05580C00 D
0193  4809 A0C0 900C C0FC          A1 R=A3                        %ISOLATE LOW ORDER 4 BITS            05590C00 D
0194  4809 A0C1 4C00 C0F0          A1 L=A1                                                             05600C00 D
0195  4809 EC41 180C C0F0          A3 + B L=A3,BAD                %MULT BY 8                           05610C00 D
0196  900C 00C0 0000 C030          COMP 3=SAR                                                          05620C00 D
0197  4809 0C41 0B0C 00F0          B L=B                          %MULT BY 2                           05630000 D
0198  8000 00C3 0C00 C030          COMP 1=SAR                                                          05640C00 D
0199  4809 EC43 0B92 C0F0          A3+D=B,MIR,INC                                                      05650C00 D
019A  8C0D 0F46 001C 00F0          IF COV THEN BMAR+1=MAR2; STEP ELSE JUMP                             05660C0C D
019B  002C 0000 0000 00EC          2=LIT                          %COUNTER SET FOR SECOND PASS FOR LOOP05670C00 D
019C  2C0B 0000 0001 00F0          IF LC1 THEN LCTR; STEP ELSE SKIP                                    05680C00 D
019D  18E0 00C0 003C 0040          DATA1-1=MPCR                                                        05690000 D
019E  4809 ACC1 40D0 00FC          A1 L=A1                        %ADD IN LSP BYTE                     05700C0C D
019F  8000 0000 00DC 0030          COMP 4=SAR                                                          05710C0C D
01A0  4809 A000 9C00 C0F0          A1 R=A3                                                             05720C00 D
01A1  4809 EC40 4B0C C0F0          A3+B=A1,B                      %COULD HAVE BEEN CALLED BY RESERVE   05730C00 D
01A2  380B 00C0 0C00 C0F0          IF LC2 THEN STEP ELSE SKIP                                          05740C00 D
01A3  0330 0000 0000 0040          RESERVE1-1=MPCR                                                     05750000 D
01A4  4809 20C3 00EC C0FC          LIT=MAR2                                                            05760C00 D
01A5  0D4C 00C3 0000 C0E0          CARDBUF=LIT                                                         05770C00 D
01A6  13EC 00C0 0000 C060          INPUT-1=CPCR                   %REREAD COLS 4-8 FOR ADDRESS         05780C00 D
01A7  113C 00C0 0000 0060          GETADDR-1=CPCR                                                      05790C00 D
01A8  4809 A000 0030 C0FC          A1=MIR                                                              05800C00 D
01A9  4809 E0C2 000C C0F0          NOT A3                         %IF ADDR=0 PUT IN LINE               05810C00 D
01AA  680B 0000 0000 C0F0          IF ABT THEN STEP ELSE SKIP %OTHERWISE PUT AT ADDRESS GIVEN          05820C0C D
01AB  0B2C 0C00 0C00 C040          NOPACK-1=MPCR                                                       0583CC00 D
01AC  4809 00C1 CB00 00FC          1 L=B                                                               05840C00 D
01AD  A00C 0000 0C0C 002C          COMP 10=SAR                                                         05850C00 D
01AE  4809 EC43 00EC C0FC          A3+B=MAR2                      %OFFESET ADDRESS BY 1024             05860C0C D
01AF  13C0 0000 0000 0060          OUTPUT1-1=CPCR                                                      05870C00 D
01B0  0E50 0000 0C00 C040          NEWWRD-1=MPCR                  %NORMAL RETURN TO GET NEXT CARD      05880C00 D
01B1  0D88 0018 00EC C0F0     IOREQ: SET GC1; WHEN GC1 THEN B111=MAR2                                  05890C00 D
01B2  13E0 C0C0 0000 0060          INPUT-1=CPCR                                                        05900C00 D
01B3  9809 0C40 CC0C 00F0          B R=A1                                                              0591CC00 D
01B4  0000 0000 0000 002C          COMP 16=SAR                                                         0592CC0C D
01B5  4809 A640 0040 00F0          A1+AMPCR=AMPCR                                                      05930C00 D
01B6  0340 00C0 0C00 00CC          EXTIOREQ-1=AMPCR                                                    05940C00 D
01B7  4809 0000 0C00 00F0          STEP                                                                05950000 D
01B8  4824 0C40 101C C0FC          B=MAR2,A3;EXEC                                                      05960C00 D
```

```
01B9  49F6 0000 0030 00FC          CALL;SET LC1                                                    0597CC00 D
                                IORETURN:                                                          0598CC00 D
01BA  4949 0019 C005 C0F0          RESET GC1; B111=CTR                                             0599CC00 D
01BB  8808 0000 0002 00F0          INC;WHEN COV THEN STEP                                          0600CC0C D
01BC  1B00 0000 0030 C04C          IOREQ-1=MPCR                                                    0601CC00 D
                                EXTIOREQ:                                                          0602CC00 D
01BD  0350 000C 0030 C0CC          DUMPREG-1=AMPCR                                                 0603CC00 D
01BE  0360 0000 CC00 00C0          DUMPADD-1=AMPCR                                                 0604CC00 D
01BF  0370 00C0 003C 00C0          CARDIO-1=AMPCR                                                  0605CC00 D
01C0  0380 0000 0C30 00C0          DISKIO-1=AMPCR                                                  0606CC00 D
01C1  0390 0003 0030 00C0          DISKWRITE-1=AMPCR                                               0607CC00 D
01C2  03A0 0000 0030 00C0          PRINTERIO-1=AMPCR                                               0608CC00 D
01C3  03B0 0000 0C00 C0C0          WORDIO-1=AMPCR                                                  0609CC00 D
01C4  03CC 0000 0000 00C0          PRINTLINE-1=AMPCR                                               0610CC0C D
01C5  000C 0000 0000 00CC          START-1=AMPCR                                                   0611CC00 D
                                DUMPREG:                                                           0612CC00 D
01C6  4809 0000 003C 00F0          STEP                          %SETUP ADDRESS OF BLANKLINE BUFFER 06130C00 D
01C7  1F9C 0000 0C30 C0C0          BLANKLINE=AMPCR                                                 0614CC0C D
01C8  4809 0003 0030 C0F0          STEP                          %ALLOW FOR TIMMING FOR AMPCR      06150C00 D
01C9  4809 0640 0030 00F0          AMPCR=MIR                                                       0616CC00 D
01CA  4809 2001 0F30 C0FC          LIT L=BBI                                                       0617CC00 D
01CB  0010 0000 0030 C0A0          16=SAR;1=LIT                                                    0618CC00 D
01CC  4809 0C40 0C9C C0F0          B=MIR                         %OUTPUT BLANKS = 30303C3C         06190C00 D
01CD  10A0 0000 0030 0060       OUTBLANK:  EXTOP-1=CPCR                                            C6200C00 D
01CE  1CCC 0000 0030 0040          OUTBLANK-1=MPCR                                                 0621CC0C D
01CF  1410 0000 0030 C040          IODUMP-1=MPCR                                                   0622CC00 D
                                DUMPADD:                                                           0623CC0C D
01D0  1410 0000 0030 C040          IODUMP-1=MPCR                                                   0624CC0C D
                                CARDIO:                                                            0625CC00 D
                                   %READ ONE CARD INTO BUFFER ADDRESSED IN B,MAR2                  0626CC00 D
01D1  4809 E0C1 1000 C0F0          A3 L=A3                       %DROP MSB OF BR REG               0627CC00 D
01D2  000C 0000 0030 C020          COMP 16=SAR                                                     0628CC00 D
01D3  2809 E0C0 8090 00F0          A3 R=MIR;IF LC1                                                 0629CC00 D
                                CARDIO1:                                                           0630CC00 D
01D4  10A0 0000 0030 006C          EXTOP-1=CPCR                                                    C631CC00 D
01D5  1D3C 0000 0030 0040          CARDIO1-1=MPCR                                                  0632CC00 D
01D6  1B90 0000 0030 0040          IORETURN-1=MPCR                                                 0633CC00 D
                                DISKWRITE:                                                         0634CC00 D
01D7  4800 0000 0000 C0F0          WAIT                                                            0635CC00 D
                                PRINTERIO:                       %ASSUME ERROR NUMBER IN A3        0636CC00 D
C1D8  4809 0003 0901 00F0          0=BC8,LCTR                                                      0637CC00 D
01D9  0210 00C0 0030 C0E0          33=LIT                                                          0638CC00 D
01DA  4809 20C3 001C C0F0          LIT=MAR2. PRINTAREA=LIT                                         0639CC0C D
01DB  0EE0 0003 0030 00E0
01DC  4812 0C40 0C90 C0F0          B=MIR;SAVE                                                      0640CC00 D
01DD  9809 0000 0030 1CF0          MW2;IF SAI                                                      0641CC00 D
01DE  9C08 0000 0C3C C0F0          WHEN SAI THEN 0;STEP                                            0642CC00 D
01DF  8429 0F46 0C1E C0FC          IF NOT COV THEN BMAR+1=MAR2,INC;JUMP                            0643CC00 D
01E0  4809 E157 180C 00F0          A3 AND LIT L=A3,BAD       %MULTIPLY ERROR NUMBER BY 10          06440C00
01E1  9070 0C00 0000 C0B0          COMP 3=SAR;7=LIT                                                0645CC00 D
01E2  4809 0C41 0B31 C0F0          B L=B,LCTR                                                      0646CC00 D
01E3  B09C 0000 C030 00B0          COMP 1=SAR;9=LIT                                                0647CC00 D
01E4  4809 EC40 100C C0F0          A3+B=A3                                                         0648CC00 D
01E5  4809 E643 401C C0F0          A3+AMPCR=MAR2,A1                                                0649CC00 D
01E6  1180 0000 0030 00C0          ERRORLIST=AMPCR                                                 0650CC0C D
01E7  4809 20C0 1030 C0F0          LIT=A3                                                          0651CC00 D
01E8  0ED0 0000 0C0C C0E0          PRINTAREA-1=LIT                                                 0652CC00 D
C1E9  13E0 0000 0C30 0060       ERROR1: INPUT-1=CPCR                                               0653CC0C D
01EA  4809 0C40 0C90 00F0          B=MIR                                                           C654CC00 D
01EB  4809 E0C3 101E C0F0          A3+1=A3,MAR2,INC                                                0655CC00 D
```

```
01EC  13CC 0000 0000 C060          OUTPUT1-1=CPCR                                                    06560C00 D
01ED  840B A0C3 40EC C0F0          IF NOT COV THEN A1+1=A1,MAR2;STEP ELSE SKIP                       06570000 D
01EE  1E80 0003 0030 004C          ERROR1-1=MPCR                                                     06580000 D
01EF  4809 00C1 0B30 C0F0          1 L=B                                                             06590000 D
01F0  0EE0 0000 003C C0A0          COMP 16=SAR;PRINTAREA=LIT                                         06600C00 D
01F1  4809 2C5C C090 00FC          LIT OR B=MIR                                                      06610000 D
01F2  10A0 00C0 0030 C06C  ERROR2: EXTOP-1=CPCR                                                      06620000 D
01F3  1F1C 0000 0070 C040          ERROR2-1=MPCR                                                     0663CC0C D
01F4  1B9C 0000 0030 00C0          IORETURN-1=MPCR                                                   06640C00 D
01F5  2829 0000 0030 00F0          IF LC1 THEN JUMP ELSE STEP                                        06650C00 D
01F6  0E50 0000 0030 004C          NEWWRD-1=MPCR                                                     06660C00 D
                   WORDIO:                                                                           0667CC00 D
01F7  13E0 0000 0030 0060          INPUT-1=CPCR                 %FETCH WORD TO BE PRINTED            0668CC00 D
01F8  4809 00D9 0C1C C0F0          B110=MAR2                    %WRITE WORD OUT TO MAILBOX+1         06690C00 D
01F9  4809 0C40 0030 C0F0          B=MIR                                                             06700C00 D
01FA  13C0 0003 0C00 0060          OUTPUT1-1=CPCR                                                    0671CC00 D
01FB  4809 E001 0B30 00F0          A3 L=B                       %GET ADDRESS INTO UPPER B            06720CC0 D
01FC  00E0 0000 0030 00AC          COMP 16=SAR;14=LIT                                                06730C00 D
01FD  4809 2C5D 8080 C0F0          LIT OR B C=MIR               %SEND 14/ADDRESS                     0674CC00 D
                   WORDIO1:                                                                          06750C0C D
01FE  10AC 0000 0C30 006C          EXTOP-1=CPCR                                                      06760C0C D
01FF  1FDC 0000 0000 0040          WORDIO1-1=MPCR                                                    06770C00 D
0200  1B9C 0C00 00DC C04C          IORETURN-1=MPCR                                                   06780C00 D
                   PRINTLINE:  %PRINT ONE LINE FROM BUFFER ADDRESSED IN B                            06790000 D
0201  2809 20C1 1000 C0FC          LIT L=A3;IF LC1                                                   06800C00 D
0202  006C 00C0 0C30 00A0          COMP 16=SAR; 6=LIT                                                0681CC00 D
0203  4809 EF4C 0030 C0F0          A3 XOR BMAR=MIR              %MSB OF BMAR ALWAYS HAS A 7          06820C00 D
0204  10A0 0003 0C00 0060 PRINTLINE1:  EXTOP-1=CPCR                                                  06830C00 D
0205  2030 00C0 0000 C040          PRINTLINE1-1=MPCR                                                 06840C00 D
0206  1B9C 0000 0030 C040          IORETURN-1=MPCR                                                   06850000 D
                   DISKIO: %READ DISK AS IN MAILBOX  B CONTAINS CODE/ADDRESS                         0686CC00 D
0207  2809 0C40 0030 00F0          B=MIR;IF LC1                                                      06870C00 D
0208  10AC 00C0 0000 0060 DISKERR: EXTOP-1=CPCR                                                      06880C00 D
0209  2070 0000 003C 004C          DISKERR-1=MPCR                                                    06890C00 D
020A  1B90 0000 0030 C040          IORETURN-1=MPCR                                                   06900000 D
020B  4849 00C0 0030 00F0 RESERVE:  SET LC2                     %RESERVE AMOUNT OF SPACE IN COLS.9-160691CC0C D
020C  18A0 0000 0C30 C040          DATA-1=MPCR                                                       06920C00 D
020D  4809 CC40 2030 00F0 RESERVE1:  A2+B=A2                    %INCREMENT PROGRAM COUNTER           06930C00 D
020E  4809 2003 0C1C CCF0          LIT=MAR2                                                          0694CC0C D
020F  0780 00C0 0000 C0E0          PLOC=LIT                                                          06950C00 D
0210  0E30 0000 0000 0040          OUT-1=MPCR                                                        06960C00 D
                   WORD:   %W - ASSUME CHARACTER INPUT UNTIL FIRST QUOTE                             06970C00 D
                           %ELSE USE ENTIRE CARD AS CHARACTER STRING. USE ADDRESS                    06980C00 D
                           %IN COLS 4-8 ELSE NEXT LOCATION IF ZERO.                                  06990C00 D
0211  1130 0000 003C 0060          GETADDR-1=CPCR                                                    07000C00 D
0212  4809 2003 401C 00F0          LIT =A1,MAR2                                                      0701CC00 D
0213  AD6C 0CC3 0030 00A0          COMP 10=SAR;COL9X12=LIT                                           07020C00 D
0214  4809 0013 8B3C 00F0          B111 R=B                                                          0703CC00 D
0215  3809 E002 0030 00F0          NOT A3; IF LC2                                                    07040C00 D
0216  6E59 C0DE 1000 C0F0          IF ABT THEN A2-1=A3; SET LC2; SKIP                                07050C0C D
0217  4809 EC40 1C30 00FC          A3 +B=A3                                                          07060C00 D
0218  13EC 0000 0030 0060 WORD1:  INPUT-1=CPCR                    %READ 4 COLS. INTO B              07070C00 D
0219  0030 0000 0C30 C0E0          3=LIT                                                             07080C00 D
021A  4812 00C0 0001 C0F0          LCTR;SAVE                    %SET UP FOR 4 CHARACTERS             0709CC00 D
021B  4809 0C41 8B30 00F0          B C=B,MIR                    %GET NEXT COLUMN                     07100C0C D
021C  03F0 0000 0030 C0B0          63=LIT;COMP 8=SAR            %MASK LEAST 6 BITS                   0711CC0C D
021D  4809 2C56 001C C0F0          LIT AND B=MAR2               %SAVE LEAST 6 BITS                   0712CC00 D
021E  4809 2F52 0030 00F0          LIT EOL BMAR                 %SEE IF ITS A QUOTE "                07130C00 D
021F  6419 0000 0032 C0F0          IF FALSE THEN INC; SKIP ELSE STEP                                 0714C00C D
0220  03D0 0000 003C C040          WORD2-1=MPCR                                                      0715CC00 D
```

```
0221  8C0D E0C0 101C C0F0          IF COV THEN A3+1=MAR2,A3; STEP ELSE JUMP                    07160C00 0
C222  3E49 E0C0 200C C0F0          IF LC2 THEN A3=A2;SET LC2  %USE NEXT ADDRESS FROM A2        07170000 0
0223  13C0 0000 0030 0060          OUTPUT1-1=CPCR                                              07180C00 0
0224  4809 A0C3 401C C0F0          A1+1=A1,MAR2                                                07190C00 0
0225  2170 0000 0C30 00CC          WORD1-1=AMPCR                                               07200C0C 0
0226  4809 A152 0C0C 00F0          A1 EQL LIT                                                  07210C00 0
C227  0E80 0000 003C CCE0          LASTCOL=LIT                                                 07220C00 0
C228  6029 0000 0030 00F0          IF FALSE THEN JUMP          %218 = LAST COLUMN OF CARD      0723CCC0 0
0229  3C09 E0C0 200C C0F0    WORD2:IF LC2 THEN A3+1=A2         %RESET PROGRAM COUNTER          07240C00 0
C22A  0E50 0000 C000 C040          NEWWRD-1=MPCR                                               07250C00 0
C22B  000C C0C0 0C0C C040          END                                                         07260C0C 0
0220  2280 00C0 0030 C040    ?END
C1C4  2000 0000 0C30 00C0
01C3  1F60 0003 C03C C0C0
01C2  1D70 0C00 003C C0C0
01C1  1D6C 0C03 0C3C 00CC
01C0  206C 0000 0C3C C0C0
01BF  1D0C 0D03 003C 00C0
01BE  1CF0 00C3 000C C0CC
01BD  1C50 00C0 0C00 00CC
01B6  1BCC 0000 0030 00C0
01A3  20C0 00C0 0030 0040
0183  1B90 0CC3 0000 C040
0175  183C 0003 0C00 0040
0170  1750 0000 0000 C0C0
0126  13E0 0000 0030 C060
0108  10AC 00C0 0C00 C060
00FF  13EC 00C3 0000 0060
00F0  10A0 0000 003C 0060
C0EB  13C0 00C0 0C3C 006C
00D2  1130 00C3 0C00 C060
00D1  1B00 0000 0030 C040
00CF  13C0 0003 003C C060
00CB  1130 00C3 0030 0060
C0CA  0E30 0003 003C C04C
00C6  1130 00C3 0C30 C060
00C5  123C 00C3 0C00 0040
00C4  113C 0003 0030 C06C
C0C3  1230 00C0 0000 0040
00C0  1130 0000 0000 006C
C0BF  0E30 0003 0C30 C040
00B9  1130 0003 003C C060
00B5  0E30 0000 0000 004C
00B0  0E30 0003 0C3C 004C
00AC  0B0C 00C0 003C 00C0
00A6  0B2C 00C3 0000 0040
00A3  0B50 00C0 0C30 006C
0090  0B5C 00C3 0000 C060
0083  0B5C 00C3 0C30 C060
0081  0A6C 00C0 0000 C04C
0075  0B5C 0003 0030 0C60
0073  0A60 0000 003C 0040
0062  0B50 00C3 0030 0060
0060  0730 0003 003C 0040
005D  073C 0003 0C30 C040
005A  0810 00C3 0C30 0040
004D  0B50 00C3 0000 0060
004A  107C 0003 0C30 C040
0049  210C 00C0 0000 0040
0046  20A0 00C3 0000 0040
```

```
0043  1400 0000 0000 C040
0040  0010 0000 0000 0040
0030  0CA0 C000 0000 0040
0030  0C50 0000 0000 C040
0037  0C30 0000 0000 0040
0034  0B80 0000 0000 0040
0031  0BF0 0000 0000 C040
002E  04A0 0000 0000 C040
002B  18A0 0000 0000 C040
0024  13E0 00C0 0000 C060
0021  0D20 0000 0000 0040
0015  10A0 00C0 0000 C060
000F  13C0 00C0 0000 C060
 0 ERRORS '   728 CARDS.   3969 TOKENS.      6700 RULES.       507 SECONDS.
# DUP FILE ON DISK:UYK7-LOADER
INVALID ARGUMENT: FILEID 4              #     1:1209.1      ##########################################################
```

APPENDIX D. SAMPLE AN/UYK-7 SORT PROGRAM

This appendix provides a copy of a sample program which was used to demonstrate the capability of the D-Machine to function as an AN/UYK-7. The program, written in AN/UYK-7 Machine Language, will read twelve numbers per card, print the numbers, sort the input and print the sorted results. Any number of cards could be read and sorted until the "END" card is encountered. At this point the program would reinitialize memory and be ready to accept the next program. Several of the Loader's macro functions, including the L, W, D and N options, were utilized along with the appropriate formats for the five instruction types. It must be emphasized that when this program was run the ALGOL Machine had been removed and the D-Machine reconfigured through microprogramming to be an AN/UYK-7. The entire output format and program execution were produced by the Loader/Emulator combination with the IOP being used strictly as an Input/Output Channel. Examples of the Loader and Debugger output optional listings for this sample program are presented in Appendix E and F.

```
L  00002       ORIGIN SORT ROUTINE AT LOCATION 0002
D  01001       10
   203320001001
   110310001002
   440310001002
   532220000014
   523220000003
   102320001000
   512420000021
   320020001000
   201320000777
   514020000002
   101310001002
   241310001001
   240310001002
   330020001000
   514020000006
   201320001000
   530000004040    JUMP TO TITLE ENDING ROUTINE
O  04000        BEGIN TITLE PRINTING ROUTINE AT 04000
   071400005000
   071400005352
   071400005352
   070400001022
   071400005044
   071400005352
   071400005110
   071400005352
   071400005154
   070410001002
   106320001002
   446320005416
   531220004060    JUMP TO COMPLETION ROUTINE IF END CARD
   071400005352
   071400005220
   071400005352
   071400005352
   071410001002
   510000000002    JUMP TO BEGINNING OF SORT ROUTINE
O  04040   RETURN HERE FROM SORT ROUTINE AND FINISH TITLES
   071400005352
   071400005352
   071400005264
   071400005352
   071400005330
   071400005352
   071400005352
   071420001002
   071400005352
   071400005352
   510000004011
O  04060   START COMPLETION ROUTINES HERE
   071400005352
   071400001002     PRINT END
   7706000          TERMINATE PROGRAM AND RESTART EMULATOR
```

```
W  05000           AN/UYK-7  EMULATION  DEMONSTRATION  PROGRAM
W  05022
W  05044      THIS IS A TEST OF THE AN/UYK-7  EMULATION
W  05066
W  05110       USING A MACHINE LANGUAGE PROGRAM
W  05132
W  05154     FOR A DEMONSTRATION SORT ROUTINE USING
W  05176
W  05220        THE FOLLOWING INPUT NUMBERS:
W  05242
W  05264      THE RESULTS OF THIS SORT
W  05306
W  05330          ARE AS FOLLOWS:
W  05352
W  05374
W  05416END
N  04000

 123 456 789 987 654 321 023 456 875 888 555 213
 357 652 389 123 583 742 928 654 987 248 965 281
 321 654 987 123 456 789 369 258 147 963 852 741
END      TERMINATE SORT ROUTINE, READY FOR NEXT PROGRAM
```

APPENDIX E. SAMPLE LOADER OUTPUT LISTING

This appendix provides a copy of the output received from the Loader when the sample program discussed in Appendix D is run. This output is obtained by turning on the IRQ switch on the Loader's Interpreter. The output serves as a hard copy program listing for the programmer and can be utilized for debugging. The address to the left of each instruction is the octal assignment for that instruction, derived from the last "O" or "L" card, and was offset by 1024. The Loader inout is terminated by an "N" card which initializes the PAR and signals the Emulator to commence execution.

```
L '00002       ORIGIN SORT ROUTINE AT LOCATION 0002
D  01001       10
2002       203320001001
2003       110310001002
2004       440310001002
2005       532220000014
2006       523220000003
2007       102320001000
2010       512420000021
2011       320020001000
2012       201320009777
2013       514020000002
2014       101310001002
2015       241310001001
2016       240310001002
2017       330020001000
2020       514020000006
2021       201320001000
2022       530000004040    JUMP TO TITLE ENDING ROUTINE
0  04000       BEGIN TITLE PRINTING ROUTINE AT 04000
6000       071400005000
6001       071400005352
6002       071400005352
6003       070430001022
6004       071400005044
6005       071400005352
6006       071400005110
6007       071400005352
6010       071400005154
6011       070410001002
6012       106320001002
6013       446320005416
6014       531220004060    JUMP TO COMPLETION ROUTINE IF END CARD
6015       071400005352
6016       071400005220
6017       071400005352
6020       071400005352
6021       071410001002
6022       510030000002    JUMP TO BEGINNING OF SORT ROUTINE
0  04040   RETURN HERE FROM SORT ROUTINE AND FINISH TITLES
6040       071400005352
6041       071400005352
6042       071400005264
6043       071400005352
6044       071400005330
6045       071400005352
6046       071400005352
6047       071420001002
6050       071400005352
6051       071400005352
6052       510030004011
```

```
0  04060   START COMPLETION ROUTINES HERE
6060       071400003352
6061       071400001002      PRINT END
6062       7706000             TERMINATE PROGRAM AND RESTART EMULATOR
W  05000            AN/UYK-7  EMULATION  DEMONSTRATION  PROGRAM
W '05022
W  05044      THIS IS A TEST OF THE AN/UYK-7  EMULATION
W '05066
W  05110        USING A MACHINE LANGUAGE PROGRAM
W '05132
W  05154     FOR A DEMONSTRATION SORT ROUTINE USING
W '05176
W '05220         THE FOLLOWING INPUT NUMBERS?
W '05242
W  05264       THE RESULTS OF THIS SORT
W  05306
W  05330           ARE AS FOLLOWS?
W '05352
W '05374
W  05416END
```

```
N 04000
          AN/UYK-7  EMULATION  DEMONSTRATION  PROGRAM

     THIS IS A TEST OF THE AN/UYK-7  EMULATION
      USING A MACHINE LANGUAGE PROGRAM
    FOR A DEMONSTRATION SORT ROUTINE USING
       THE FOLLOWING INPUT NUMBERS?

 123 456 789 987 654 321 023 456 875 888 555 213

    THE RESULTS OF THIS SORT
        ARE AS FOLLOWS?

 023 123 213 321 456 456 555 654 789 875 888 987


       THE FOLLOWING INPUT NUMBERS?

 357 652 389 123 583 742 928 654 987 248 965 281

     THE RESULTS OF THIS SORT
        ARE AS FOLLOWS?

 123 248 281 357 389 583 652 654 742 928 965 987


       THE FOLLOWING INPUT NUMBERS?

 321 654 987 123 456 789 369 258 147 963 852 741

    THE RESULTS OF THIS SORT
        ARE AS FOLLOWS?

 123 147 258 321 369 456 654 741 789 852 963 987


END     TERMINATE SORT ROUTINE, READY FOR NEXT PROGRAM
```

APPENDIX F. SAMPLE DEBUGGER OUTPUT LISTING

This appendix provides a sample of the output received from the Loader when it was used as a debugger for the sample program discussed in Appendix D. This output is obtained by turning on the IRQ switch on the Emulator's Interpreter. The switch was not left set for the entire program's execution because the output requires approximately thirty pages. The IRQ switch causes all CMR registers from address 0000 to 0035 to be printed as each instruction is executed. Each time and instruction does a Y-Operand write that address is also dumped. The addresses are printed in the leftmost column and include the following information which can be used during program debugging:

OCTAL ADDRESS	DESCRIPTION
0-7	Task A-Register contents
10	Unused (always = 0)
11-17	Task Index B-Register contents
20-27	Task Base S-Register contents
30	Repeat Instruction (if applicable)
31	Current Instruction being Repeated (if applicable)
32	Current Indirect Control Word (if applicable)
33	Y-Operand for Indirection (if applicable)
34	ICW address in memory (if applicable)
35	Program Address Register

Those addresses marked "if applicable" would normally be zero, but could contain data if either the Repeat mode or

Indirection mode had been used previously in the program. The lower 20 bits of address 0035 (PAR) point to the octal address of the next instruction. The upper 12 bits of the PAR are those ASR bits which were implemented as discussed in Section C of Chapter V. It is noteworthy that the output is an eleven bit octal representation of a 32-bit binary word, thus the leftmost octal digit represents only two bits.

By utilizing this optional listing in combination with the Loader listing, the programmer should be capable of proceeding through his program step by step. This facility proved an invaluable aid in troubleshooting the Emulator as each new instruction was written and tested.

```
0C0000C0    C3C0C00C0000    00000000000    00000000000    C0000C000C00    00000C000000    C0C0C0000000
0C000006    C3D000000C00    00000000000    00000000000    C0000C000C00    00000C000C00    C000C0C0C000
C0000C14    03C00000CC30    00000030000    00000000C00    00C0C0000C00    00CCC002C0C    C0CCC022CCC
0C000022    03000042C030    00000062C00    00C00102C00    C0000122C0C    C0CCC142C00    C0000162C000
0C000030    C3C000000C00    00000300000    C0000000C000    C000C0000C00    00000C000C00    C0C0C0C6000
          AN/UYK-7  EMULATION  DEMONSTRATION  PROGRAM

0CD0D000    03000C000C00    C0000C000000    C0000000C000    C0000C000000    00000C000CC0    C0C0C0C00000
0C000006    C3C000000C00    C0000300C0C0    0000000C0C0    00000C000000    00C0C000C00    C0C0C300C0C
0C000014    03D000000C00    0000C000C00    C0C0000C0C00    C000000C0C00    0000C002000    C0C0C022000
00000C22    03C0C042C030    00000062C00    00000102000    C0000122C00    C0000142CC0    C0C0C162CCC
0C000C30    03000C000C00    C0C000000000    00C00000000    C0000000000    C0000000C000    CCC0C006C01

CC00D000    C3C0C00C0C00    C0000000C00    C0000C000000    C0000C000000    00000C000000    CCC0C0000C00
00000006    03000C00CC00    00000300000    00000000C00    C000CC000000    C0000C00000C    C0CCC0000C0
0C000014    03C0C0000000    00000000000    C0000000000    C000CC000000    00000C002000    C0CCC022CCC
00D00022    03000042C030    C0000062000    00000102000    C0000122C00    C000C142C0C    C0000162C0C
0C000030    03C000000C00    00000300000    00000000000    C000C0000000    C0C0C0000000    C0CCC006002

0CD00000    C3C0C0000030    00000000000    00000000000    C0000C00C00    00000C0000C0    CCC0C0C0C00
0C000C06    03000000C000    00000000000    C0000000C00    C0000C00C000    C000C00CC00    C0CCC0C0C00
0C000014    C3000C00C030    00000000C00    00C00000000    C000000000C0    00000C002000    C0CCC022C0C
0C000022    03C0C042C030    00000062000    C0000102000    00000122000    00000142CC0    C0CCC162C0C
0CD00030    03000000CC00    00000300000    00000000000    C0000C000000    C00CC0C00000    C0000C06C03

0C000000    03000000C030    00000000000    0C00000C0000    C000C000C00    000CC000C00    C000C000C0C
0C000006    03000000C030    00000300C00    C0000000000    C000C000000    C000C000C0C    C0C0CC00C00
0C000014    03D0C000C000    00000000000    C000C000300    C000C000000    00000C002000    C0CCC022C0C
0CD00C22    03D0C042C030    00030062000    00000102000    C3000122000    C000C142C0C    C0CCC162C0C
0C000030    03000000C00    00000000000    C0000000000    C000C000C00    C0CCC0C0C0C0    C0CCC006C04
     THIS IS A TEST OF THE AN/UYK-7  EMULATION

CC000030    C3000000CC30    C000C000000    000000000C00    C0000000000    000CC000000C    C0C0C0000000
CC000006    03000000CC00    C0000300000    00C000000C00    C0C0C000000    000CC000C00    C0C0C0C0000
0C000C14    C3000000CC00    00000000C00    C0C0C0000C00    C000C000C00    C0CCC002C00    C0C0C022C00
0CD0D022    C3D0C042C030    00000062000    00000102000    C0000122C00    000CC142C0C    C0CCC162000C
0C000C30    03D0C00CCC00    C0000000C00    C0000000000    C000C000000    00000C000000    C000CC006CC5

CC00DC00    03D000000C00    00000000000    C0000000000C    C000C000C00    C000C000C00    CC0000000C0C
0C00DC06    03000000C00    0000C000C00    C000C000000    C000C000C00    C000C000000    C000C000000
0C00DC14    03D0C000003D    00000000000    C0C000000C00    00000000C00    C0C0C002C0C    C0C0C022C0C
0CD00022    03C000042C30    C0000062000    00000102000    C0C000122CC0    00000142C00    C0CCC162CC0
CC000C30    03000000C00    00000000000    00000000C00    C000C000000    00C0C000C00    C0CCC006C06
       USING A MACHINE LANGUAGE PROGRAM

0C000000    03C0C0CCCC00    00000000000    C0000C000000    C000CC000000    00C0000CCC00    C000C000C0C
0C000C06    03C0C0000C30    0000C000000    00CCC0000C00    00C0C000000    000CC0C0C00    C000C000000
0C00D014    03D0C0000000    00000000000    C0CC0000000    C000C002C00    000CC002C00    C0CCC022000
0CD00022    03CCC042C030    00000062000    C000C102C00    C30C0122C00    C0000142C00    C0CC0162000
0CD00030    03000C000C00    0000C000000    C000000C0000    C000C000C00    C0000C000C0C    C0C0C0C06007

0CD00000    03000000C030    00000300000    C0000C000000    C000C000C000    0000C000000C    C0C0C0000C0C
0CD0D006    03000C000C00    00000000000    0000000C0C00    C0000000C00    0000C000000C    C0CCC0C0C00
00000014    C3000C00CC30    0000C300C00    C3C00000C000    0000000000C0    C000C002C0C    C0CCC022C0C
0C0000C22    03000C042C30    00000062C00    C0000C102000    C0000C122C00    C000CC142000    C0CCC162C30
0C000030    C3000000C000    00000000000    C0C000000000    C000C0000C00    0000C0000C00    C000C0C6C10
     FOR A DEMONSTRATION SORT ROUTINE USING
```

```
0C0D0000   0300C00CC00   0000000000C0   C0C000000C00   C000C000000   00000000000   CCCCCCCCCCC
0C000006   03C000000C00   0000000000C0   0000000C00C   C000C000000   C00CC000000   CCCCCCCCCCC
0C000014   0300000000C0   0000000000C0   C000000C00C   0000C000000   C0C0C002C00   CCCCCC22000
0C000022   0300C042C00   0000006200C   C0000102000   C0000122C00   0000C142000   C000C162CCC
0C000030   0300C000C00   0000000000C   0000000C000   C0000000000   0000C000000   CCCCCC06C11

CCD00C00   C300C000C00   C0000000000   C000000CC00   C000C000C00   C00CC000000   C0CC0000C00
00000006   03C000000C0   0000000000C   0000000C00C   C00C0000C00   C00CC000C0C   CCCCC00C000
CCD00014   03C000000C0   C0000000C0C   0000C000C0C   C000C000C0C   C00C0002C00   C0000022C00
0C000022   C3000042C00   C000006200C   C0000102C00   C000C122C00   0C000142C00   C0CCC162CCC
0CD00030   0300000C0C0   00C000000C0   C000000000C   C000C000C00   0000C00CC00   C0C0C006012

0C000000   03C0C0000C0   0000C000000   0000C000C00   C000C0CC0C0   0000C000C0C   CC0CC000CCC
C0000006   C5000201C03   0000030000C   C000000000C   C00C0000C0C   0000C000C00   CCCCC000000
0C000014   03C000000C0   0000000C00C   C0C000000C0   C000C0C0C0C   000CC002CCC   C0000022C00
0C000022   C3000042C30   C0000062C00   C0000102C0C   C0000122C00   000CC142C00   C0CCC162C0C
0C000030   03C0C000C00   0000000000C   0000C000C00   C0000000000   0000C000000   C0C0C006013

0C000000   03C000000C0   C000000000C   C000000C00C   C00CC000000   CCC0C000C00   CCCCC00CC0C
00000C06   0503C201C03   0000000000C   C000C000C0C   C0C00C000C0   0000C000C00   CCCCC0C0C0C
0C000014   03C000000C0   0000000000C   0000000000C   C00CC000000   C0000C002000   C0CCC022C0C
0C000022   0300004200C   0000006200C   0000010200C   C0000122000   C0C0C142000   C0CCC162CCC
0C000030   0300C000000   0000000000C   0000000000C   000CC000000   C00CC000000   10C00006014

0C000000   03C0C000C00   0000C000000   0000C0C0000   C000C000C00   0000C00CC0C   C000C0C0CC0
0C000006   0500020100C   0000030000C   C0C000000C0   C00C0000C0C   0000C000C00   C0C0C00CC00
0C000014   03C000000C0   0000C000000   C0C000000C0   C000C000C00   C000C002000   C0C0C022000
0CD00022   C3C0C042C00   0000006200C   0000C102000   0000C122C00   0000C142C00   C0CC0162C0C
0C000030   0300C000000   0000000000C   0000000000C   C000C000CC0   0000C000C00   1000F0C6015

0C000000   0300000C000   0000C000000   0000000C00C   C000C000000   C0CCC0C0000   CCCCC000C0C
0C000006   C5000201003   0000000000C   C0C000000C0   0000C000000   0000C000000   CCCCC0CCC0C
0CD00014   0300000C00C   0000000000C   0000000C00C   C0000002000   0000C002000   C0CCCC22000
0CD00022   0300004200C   0000006200C   0000010200C   C300C122000   C000C142C00   CCCCC162C0C
0C000030   C300000CC00   C000C000000   000000000C0   C00CC000000   C000C0C0C00   1000F006016
        THE FOLLOWING INPUT NUMBERS?

0C000000   03C000000C00   0000C000000   0000000000C0   C000C000000   0000C000C00   CCC0C000C00
0C000006   0500020103   0000000000C   0000000000C   C00CC000000   C000C0C0C30   C0C0C0C0C0C
0C000014   C3000000C30   0000C000C00   0000000000C   C000C000C00   000CC002C0C   CCCCC022C0C
CCD00022   0300004200C   0000006200C   0000010200C   C000C122C00   C000C142C00   C0C0C162CCC
0C000030   C3C0C000000   0000000000C   0000000000C   C000C000000   C000C000000   1000F006017

0C000000   03C000000C00   0000000000C   CCC000000C0   C000C000000   C000C000C0C   CCCCCCC0000
0C000006   0500020103   C0000000000   C0C00000C00   0000C000000   C000C000C0C   C000C000C00
0CD00014   0300000CC00   C000000000C   C000000C000   C000C000000   C000C002C0C   CCC0C022CCC
CCD00022   03C0004200C   C0000062000   C000C102000   C000C122C00   0000C142C0C   CCCCC162000
0C000030   0300000C00   0000000000C   0000C000000   C0000000000   0000C000C00   10C00006C2C

00D00000   0300C00CC30   0000000000C   00C000000C0   C000C000C00   C0CCC00CC00   00C0C000CC0
CCD00C06   C5CCC201C03   C000C300000   C000000C00   C00CC0C0000   00C0C00000C   C0C0C000000
00D00014   03CC000CC00   0000000000C   C0C0C000C0C   C00C0002000   C000C022C00   C0C0C022000
CCD00022   C3C0C042C00   C0000062000   0000010200   0000C122000   C0000142C00   CC00C162C0C
0CD00030   C300000C00   C0000000000   0000000000C   C000C9C0C00   C00CC000000   100CC006021
 123 456 789 987 654 321 C23 456 875 888 555 213

0CD00000   C3000000CC30   C0000000000   C000C000C00   C000C000000   C000C000C00   CCCCC0C0000
0CD00C06   0500C201C03   0000C000C00   C0C000000C0   0000C000000   0000C0C0C0C   C0CCC000C00
CCD00014   C3000C0C030   0000000000C   C0C0000000C   C0000000000   00C0C002000   CCCCC022000
0C000022   C3C000042C00   0000006200C   C0C0C102C00   C000C122000   C000C142C00   C0CCC162C00
0CD00030   0000C00CC30   0000000000C   0000000000C   C000C00C000   00CCC000C00   10C0C006022
```

0C000006	C5C00201C03	000000000C00	00C000000000	C000C000000	C000CC00C0C	CCCCC00C000
0CD00014	03000000CC00	00000000C000	000000000C00	C000C000000	C000C002C00	C0C0C022000
0C000022	0D000042C00	00000062000	C0C001020C0	C000C122000	0000C142C00	C0CC0162C0C
0C000030	C300000C0000	0000C000000	00000000000	C000C000000	C00CC000000	10CCC002C02
0C000000	C3C000000C00	000003000000	000000C0CC00	00000C000000	C0C0C000000C	C0CCC000C000
0CD00006	05C00201C03	00000C000C00	C0C0C0000C00	C000C0000000	C00CC0C0C00	C0CCC000C12
00000014	C3000000CC00	00000000000	C0000000000	C000C000C00	00C0C002C00	C0CCC022000
0CD000C22	03000042C00	C0000062000	C0C00102C30	C000C122C00	C000C142000	C0CCC162C00
0C000C30	C3C0C0000C00	C0000000000	C0000C000000	C0000000C00	C000C0C00000	10000C002C03
0C000C00	05C00201C03	000000000000	C000C000C00	C000C000C00	C0000C000000	C00C0CCCC00
0C000006	05000201C03	C0000000000	C0C0C000C00	C000CC00C001	C000C00C000	CCC0C0C0012
0C000C14	C3000000000	C00C0000000	C0C000000C00	000CC000C00	C0CCC002C0C	CCC0C022000
0C000C22	C3000042C00	00000062C00	C0C00102C30	C000C122C00	C000C142000	C0CCC162000
0C000C30	C3C0C000C00	00000000C00	C0000000000	C000C000C00	0000C000000	10CCC002C04
CC000000	05CCC201C03	000000000000	C000C000C00	C000C000000	C000C000000	CCC0C0C0C00
00000006	05000201003	000000000000	C0000000000	C000CC00001	C00CC000000	C0CCC000012
0C000014	C3000000000	C000C000000	0C0C0000C00	C000C000000	00C0C002C00	CCCCC022000
CC000022	03000042C00	00000062000	00000102000	C3000122000	C000C142CC0	C000C162C90
0CD000C30	C3C000000C00	000003000000	C000000000C0	000000000C00	C000C0CCC0C	C0C0C002005
0CD00000	05000201C03	000003000000	C000000000C0	C000C0C0000	C00CC0C0000	CCC0CC00C0C
0C000006	05C00201C03	000000000C00	C000000000C0	C000C000C01	0000C0C0000	CCC0C000012
00D00014	C3C000000C00	000000000000	C0C000000C00	C00CC000C00	0000C002000	C0C0C022000
0C000022	C3CCC042C00	00000062000	00000102C0C	C0000122000	C0C0C142000	C000C162000
0C000030	C3C0C00CC00	000000000000	C000000000C0	C000C300000	0000C0C0000	C0C0C002006
0C000000	05000201C03	000000000000	C0C000000000	0000C000000	00000C0C0000	CCC0C0000C00
CC000006	05000201C03	00000C00C000	C0C000000000	C0000000001	000C0000000	C0C0CC000C11
CC000014	C3C0C000C00	000000000C00	00C000000C00	C000C002000	C000C002000	C0CCC022CCC
0C000C22	03000042C00	00000062000	C0000102000	C0000122C00	0000C142C00	CCC0C162C0C
0C000030	C3C0C000CC00	000000000000	C0000000C30	C0000000C00	0000C000C0C	C000C002C03
0C000C00	C5001002406	000000000C00	00C0C0000000	C000C0000000	000CC000C00	C0C0000C0CC0
0CD00006	05000201C03	000000000C00	00C000000C0C	C00CC000002	000CC000000	CCC0C0000C11
0C000014	C3C0C000CC00	C0000000000	C00C0000C30	C00CC000000	C000CC02000	CCCCC022C00
0C000C22	03C00042C00	00000062C00	0000C102000	C00C0122000	C0C0C142000	C000C162000
CCD00030	C3C0C0000000	000003000000	C000C000000	C300C000C0C	C000C000C00	CCC0C002C04
CC000000	05001002406	000000000000	C00000000C00	C000C3000C00	0000C0C0C00	C0C0C000C00
0C000006	05C00201C03	000003000C00	C00000C00C00	C00000000C2	C00CC000C0C	C0CCC0000C11
00000014	C3000000CC00	000000000000	C000000000C0	C000C002000	C000C002000	CCCCC022C00
0CD00022	03000042C00	00000062000	C0C00102000	C300C122000	C00CC142000	C0C0C162CCC
0C000030	C3C0C000CC00	000000000000	00C000000000	C00C0000000	C0C0C0C0C00	C0C0CC02C05
0CD00C00	05001002406	000000000C00	C0000000000	C00000000000	C0000C000000	C0CCC3C0000
0CDC0006	C5C00201003	00000C000000	C0000000C00	C0000C00002	0000C000C0C	C00CC000C11
0C000014	C3C0C000CC00	000000000C00	C000000000C0	C000C000C00	000CC002C00	CCCCC022C00
0C000C22	C3000C042C00	00000062000	0000C102000	C0000122C30	CCCCC142000	C0C0C162000
0C000030	C3C0C000C000	000C00000C00	C000000000C0	C00CC000C00	C00CC000000C	C00000002006
00000000	C5001002406	0000C000C00	C000C000CC00	C000C000000	C0C0C3C0C0C0	C0C0C0C0C00
C0D00C06	C5000C201C03	00000C000000	C000C000C00	C0000000002	C000C000C0C	CCC0C000C010
0CD00014	03000C000C30	000003000C00	0000C000030	C00CC300000	C000C002C00	CCCCC022C0C
0C000C22	03C0C042C00	C0000062000	C0C0C102000	C3C0C122C00	000CC142C00	C0CC0162C0C
0C000030	03C0C0CCC00	000000000000	00000000000C	C000C000C000	0000C000C0C	C000C0C2003
0C000000	C5001604011	000000000000	C0000000000	C00CC0C0C00	0000C000C0C	C0C0C3C0000
0CD00006	05000201C03	C0000000000	C0C0000000C0	C000C300C03	C0C0CC000C0	C0CCCCC0C10
0CD00014	C3000000C000	000000000000	0000000CCC00	C000C000C30	000CC002C00	C0C0C022CCC
0CD00022	C3CCC042C00	00000062000	000000102000	C300C122C00	C000C142000	CCCCC162CCC
0C000C30	03C0C0000C00	00000C000C00	C0000000C0C	C000C000C00	C00CC000C00	C0C0C002C04

```
0CD000C06        05C00201C03        00000C000000        C0000C000000        00000C000003        00000C000C00        C0C00000C10
00000014         03C0C0000C00       C0000000000         C0000000C00         C0000C000000        C0C0C002C0C         C0C0C022C0C
0CD000022        03000042C00        00000062000         C0000102C00         C0000C122C00        C0000C142000        C0CCC162000
0CD000C30        C3000C000C00       00000000000         00C00000000         C0C0C000000         C0000C000000        C0000C002C05

0CD0D0000        05001604011        00000000000         C0000000C30         C0000C000C00        C0000C0C0000        C0000C000CC
0CD000006        05000201003        00000000000         C0C0000C0C00        0000C000003         00000C000000        C0CCC000010
0CD00014         03000000C00        C0000000000         00C00000030         C0000C000000        C00CCC02C00         00CCC022C00
CC000C22         C3000C042C00       00000062000         0000C102000         C0000C122C00        C0000C14200C        C0C00162C00
0CD000C30        0000C000C00        00000000000         00000000000         C0000C0C0000        C0000C000C0C        C0CCC002C0C
```

THE RESULTS OF THIS SORT

ARE AS FOLLOWS?

C23 123 213 321 456 456 555 654 789 875 898 987

THE FOLLOWING INPUT NUMBERS?

357 652 389 123 583 742 928 654 987 248 965 281

THE RESULTS OF THIS SORT

ARE AS FOLLOWS?

123 248 281 357 389 583 652 654 742 928 965 987

THE FOLLOWING INPUT NUMBERS?

321 654 987 123 456 789 369 258 147 963 852 741

THE RESULTS OF THIS SORT

ARE AS FOLLOWS?

123 147 258 321 369 456 654 741 789 852 963 987

END TERMINATE SORT ROUTINE, READY FOR NEXT PROGRAM

GLOSSARY

Active Status Register (ASR): A special word in the AN/UYK-7 used to indicate the status of special conditions or modes of operation. Each bit of the word has a separate meaning to the central processor.

ADO: The Burrough's Advanced Design Organization, Paoli, Pennsylvania, which designed the Naval Postgraduate School's D-Machine.

A Registers (A1, A2, A3): Each of the three A registers is functionally identical. The A registers are used for temporary data storage within the Logic Unit of the Interpreter and serve as a primary input to the adder.

Adder: The adder in the Logic Unit of the Interpreter, is a modified version of a straightforward carry lookahead adder. It is also used for executing logic operations.

Alternate Microprogram Count Register (AMPCR): The AMPCR is a 12-bit register in the Memory Control Unit of the Interpreter, which contains the jump or return address for program jumps and subroutine returns within a microprogram.

AMPCR: Alternate Microprogram Count Register.

Arithmetic (A) Registers: There are two sets of eight Arithmetic Registers in the AN/UYK-7. They are used extensively in arithmetic calculations and are directly addressable by the programmer.

ASR: The Active Status Word in the AN/UYK-7.

B Register: The B register is the primary interface between the Logic Unit of the Interpreter and the Data/Program Memory or Devices through the Switch Interlock. It also serves as a secondary input to the adder.

Barrel Switch: The barrel switch is a matrix of gates in the Logic Unit of the Interpreter, used to shift a parallel data word any number of places to the left or right in a single clock time.

Base Registers 1 and 2 (BR1, BR2): The Base Registers are two 8-bit registers in the Memory Control Unit of the Interpreter, which usually contain the base address of a 256-word block of Data/Program Memory.

Base (S) Registers: There are two sets of eight Base Registers in the AN/UYK-7 used extensively in multiprogramming and multi-processing. Base registers are used for operand address calculations.

BR1 and BR2: Base Registers 1 and 2 in the Interpreter.

Building Block: The term used to describe the primary functional units of the Interpreter Based System and includes: Interpreter, Data/Program Memory and Switch Interlock.

Control Memory Registers (CMR): A block of 89 registers in the AN/UYK-7 used for special fast memory operations.

Condition Register (COND): The COND is a 12-bit register in the Control Unit of the Interpreter and is used to store various condition bits for use during program execution.

Central Processing Unit (CPU): The primary arithmetic and control unit in a conventional computer system.

Condition Select: The condition select is a matrix of gates in the Control Unit of the Interpreter that computes the results of a computation or logic operation in the Logic Unit with a preselected result. The results of the comparison may be used to determine the sequence of execution of microprogram instructions.

Control Unit (CU): The CU, one of the five functional units of the Interpreter, is used for condition testing and the storage and distribution of enable signals received from the nanoinstructions.

Counter (CTR): The CTR is an 8-bit counter in the Z register section of the Memory Control Unit of the Interpreter, used for loop control and other counting functions.

CTR: Counter in the Interpreter.

Data/Program Memory: The Data/Program Memory of the Interpreter, also called S-Memory, provides storage for data and program, and functions similarly to the main memory modules of a conventional computer system.

Emulation: Describes a process through which the hardware components of one machine (Host) are made to "appear" to assume the specific characteristics of the hardware of another machine (target).

End-Around-Shift: A shift operation in either the left or right direction, in which the bit or bits which would be shifted out of the register are reinserted in the more significant end.

End-Off-Shift: A shift operation in either the left or right direction, in which the bit or bits shifted out of the register are lost. Vacated bit positions are automatically replaced by zeros.

Firmware: In the Interpreter Based System, firmware is the combination of stored logic in the micro-memory and the hardware logic of the Interpreter.

GC Bits: A set of two (GC1, GC2) global condition bits in the Interpreter. They are used as lockouts during communications to the I/O processor.

HALFFETCH: A subroutine written for the Emulator, which is executed when a halfword instruction has been decoded.

Horizontal: A type of microprogramming instuction which controls multiple gates simultaneously allowing parallel functions to execute.

Host: A term used in Emulation to define the machine on which another machine is to be emulated (imitated). In this thesis the Burrough's D-Machine.

ICW: Indirect Control Word in the AN/UYK-7.

IFETCH: A subroutine written for the Emulator, which reads the next instruction from main memory, deciphers the Op-code and passes control to the appropriate Op-code execution subroutine.

Incrementer (INCR): The INCR is in the Memory Control Unit of the Interpreter and increments the address of the next microinstruction to be executed by the Interpreter by zero, one or two, depending on the successor instruction which is either implied or specified. A WAIT causes an increment by zero, a STEP causes an increment by one, and a RETN causes an increment by two.

Indirection: An addressing technique or mode of operation of the AN/UYK-7 in which Y-Operand address calculation points to an Indirect Control Word (ICW) which in turn points to the operand or another ICW. Indirection is determined by the i-field of an instruction.

Indirect Control Word (ICW): The ICW is a 32-bit word in the AN/UYK-7 used in Indirection which can be broken down into several fields. The fields determine the mode of indirect address calculation to be used in pointing to the next ICW or the Y-Operand.

Indexing: An addressing technique wherein the normal address calculated is incremented/decremented (indexed) by the value in the specified index register.

Index (B) Register: There are two sets of seven Index Registers in the AN/UYK-7. They provide the ability to perform Indexing during memory referencing, and may also be used as counters.

INT: An Interrupt signal issued by an Interpreter.

Interpreter: The Interpreter is the basic building block of the Interpreter-Based System. Functionally, it is characterized by the combination of microprogram instructions stored in its M memory and the combination of bits in its nanoinstruction which enable signals to implement the hardware logic.

Interpreter-Based System: A computer design concept that provides, in the form of basic building blocks, the throughput and flexibility for a variety of data processing requirements.

Interrupt: In the AN/UYK-7, the Interrupt signals the central processor to cease its normal instruction flow and respond to a request for services. In the D-Machine, Interrupt (INT) is a flag which may be set between processors signalling that a message has been placed in the Mailbox.

Least Significant Bit (LSB): For a number or value represented in binary notation, that bit position which represents the least significant portion of the number.

LIT: Literal Register in the Interpreter.

Literal Register (LIT): An 8-bit register in the Z section of the Memory Control Unit of the Interpreter, which is used for temporary storage of literals from microinstructions.

Logic Unit (LU): The LU is one of the five major functional units of the Interpreter. It performs all of the arithmetic, Boolean logic, and shifting operations of the Interpreter.

Mailbox: An address (64K) in the Interpreter's S-Memory which is used for passing messages between Interpreters and the IOP.

MAR: Memory Address Register in the Interpreter.

Memory Address Register (MAR): The MAR is an 8-bit register in the Memory Control Unit of the Interpreter, which contains the least significant 8 bits of a memory or device address.

Memory Control Unit (MCU): The MCU is one of the five major functional units of the Interpreter. It controls the sequence of execution for microinstructions; the addressing of Data/Program Memory; and the selection of devices.

Memory Information Register (MIR): The MIR is a register in the Logic Unit of the Interpreter which serves as the output interface register between the Interpreter and the Switch Interlock.

Microprogram Address Control Register (MPAD CNTL): The MPAD CNTL, a register in the Memory Control Unit of the Interpreter - controls the loading of the MPCR and the AMPCR, and determines the value of the increment.

Microprogram Count Register (MPCR): The MPCR, located in the Memory Control Unit of the Interpreter, is a 12-bit register that usually contains the address, in M-memory, of the microinstruction presently being executed by the Interpreter.

Microprogram Memory (M-Memory): The M-Memory is one of the five major functional units of the Interpreter. It stores microinstructions which characterize the Interpreter for a given application, and may be implemented as a read/write semiconductor memory.

Microprogramming: A technique for implementing the control functions of a digital computer using programmable control signals in a separate memory called a control store, which is organized on a word basis.

MIR: Memory Information Register in the Interpreter.

Monoprogramming: A general computer operating environment in which one program at a time is executed.

Most Significant Bit (MSB): For a number or value represented in binary notation, that bit position which represents the most significant portion of the number, or the sign of the number.

MPCR: The Microprogram Count Register in the Interpreter.

Multiprogramming: A general computer operating environment in which more than one program at a time is executed with each given a fixed time slice.

Multiprocessing: A general computer operating environment in which two or more central processors execute simultaneously, and share resources; such as, memory and I/O devices.

Multiprocessor: A network of computers capable of simultaneously executing two or more programs or sequences of instructions by means of multiprogramming, parallel processing or both.

Nanoinstruction: A single instruction stored in N-Memory of the Interpreter, the contents of which constitue 56 unique signals for controlling hardware logic of the Interpreter.

Nanomemory (N-Memory): The N-Memory, one of the five functional units of the Interpreter, stores 56 specific enable signals for the hardware logic within the Logic Unit, Control Unit, and Memory Control Unit.

Page: A grouping of addresses of fixed length. In the Emulator the AN/UYK-7 memory was treated as eight pages of 8K each.

PAR: Program Address Register in the AN/UYK-7.

Program Address Register (PAR): A register in the AN/UYK-7 which holds the address of the next instruction to be accessed.

Port Select Unit (PSU): The PSU provides control and the electrical interface between a single Interpreter and its Devices and Data/Program Memory.

Shift Amount Register (SAR): The SAR is a 6-bit register in the Control Unit of the Interpreter and is used to store the number of positions a word or literal is to be shifted by the barrel switch.

Switch Interlock (SWI): The SWI provides the interconnection between Interpreters, Data/Program Memory, and Devices of an Interpreter Based System. Its function is to permit any one of a multiplicity of Interpreters to access all modules of an array of Data/Program Memory and/or all Devices.

Target: A term used in Emulation to define a machine to be emulated (imitated). In this thesis, the AN/UYK-7.

TRANSLANG: A computer programming language designed to convert pseudo-English language statements defining the action of the Interpreter for each machine cycle into binary patterns for the M-Memories.

Vertical: A microprogramming instruction which controls those gates needed for a single function execution.

Z Register Section: A collection of registers and selection gates in the Memory Control Unit of the Interpreter, which includes the CTR, LIT, and Input Selection gates used to control the execution sequence of microinstructions.

BIBLIOGRAPHY

1. Agrawala, A. K. and Rausher, T. G., "Microprogramming: Perspective and Status," IEEE Trans, C23, p. 817-37, August 1974.

2. Allred, G. R., "System/370 Integrated Emulation Under OS and DOS," Proceedings SJCC, 38, p. 163-67, 1971.

3. Almes, G. T., Drongowski, P. J., and Fuller, S. H., "Emulating the Nova on the PDP 11/40: A Case Study," Computer Conference, p. 53-6, Fall 1975.

4. Bagley, J. D., "Microprogrammable Virtual Machines," Computer, p. 38-42, February 1976.

5. Boulaye, G. and Mermet, J., Microprogramming, Herman, Paris, 1972.

6. Bricker, G. B., "Taking the Risk Out of System Upgrading," Data Processing Magazine, 12, p. 27-9, September 1970.

7. Burrough's Corporation Report 64116, Emulation Facility, by Advanced Design Organization, Paoli, Pa., 28 June 1971.

8. Burrough's Corporation Report TR70-2, The Interpreter, by R. L. Davis, 16 February 1970.

9. Burrough's Corporation Report TR70-8, Microprogramming Manual for Interpreter Based Systems, by Advanced Design Organization, Paoli, Pa., November 1970.

10. Burrough's Corporation Report 66143, Algol Reference Manual for the Interpreter Based System, by Advanced Design Organization, Paoli, Pa., 15 June 1975.

11. Dolhoff, T. L., "The Negative Aspects of Microprogramming," Datamation, 20, p. 64-6, July 1974.

12. Ellenby, J., "Emulation and Competition in I/O System Design," Computer Conference, p. 303-6, 1974.

13. Flynn, M. J., Neuhauser, C., and McClure, R. M., "EMMY-An Emulation System for User Microprogramming," Proceedings NCC, 44, p. 85-9, 1975.

14. Galey, J. M., "Microprogramming: The Bridge Between Hardware and Software," Computer, p. 23, August 1975.

15. Husson, S. S., Microprogramming Principles and Practices, Prentice-Hall, Inc., 1970.

16. Jaeger, R., "Microprogramming: A General Design Tool," Computer Design, 13, p. 150-7, August 1974.

17. Jones, L. H., "Instruction Sequencing in Microprogrammed Computers," Proceedings NCC, 44, p. 91-8, 1975.

18. Jones, L., "A Survey of Current Work in Microprogramming," Computer, p. 33-8, August 1975.

19. Jones, L. H. and Merwin, R. E., "Trends in Microprogramming. A Second Reading," IEEE Trans, C23, p. 754-9, August 1974.

20. Kahn, P. G. and Fuller, M. E., "Program Conversion: A Discussion of Techniques," Data Processing Magazine, 11, p. 28-31, November 1969.

21. Mallach, E. G., "Emulator Architecture," Computer, p. 24-32, August 1975.

22. Mandell, R. L., "Hardware/Software Trade-Offs--Reasons and Directions," Proceedings FJCC, 41, p. 453-9, 1972.

23. Rauscher, T. G., "On the Feasibility of Emulating the AN/UYK-7 Computer on the AADC Signal Processing Element," NTIS, November 1972.

24. Reigel, E. W. V. and Fisher D. A., "The Interpreter--A Microprogrammable Building Block System," Proceedings SJCC, 40, p. 705-23, 1972.

25. Rosin, R. F., "Contemporay Concepts of Microprogramming and Emulation," Computer Survey, 1, p. 197-212, December 1969.

26. Sperry Rand, UNIVAC, Computer Set AN/UYK-7 (V) Technical Manual Volume 1, Navships 0967-319-4010, January 1971.

27. Sperry Rand, UNIVAC, AN/UYK-7, Technical Description.

28. Tucker, A. B. and Flynn, M. J., "Dynamic Microprogramming: Processor Organization and Programming," Communications of the ACM, 14, p. 240-50, April 1971.

29. Wilkes, M. B. "The Growth of Interest in Microprogramming: A Literature Survey," Computer Survey, 1, p. 139-45, September 1969.

INITIAL DISTRIBUTION

	No. Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia, 22314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Department Chairman, Code 52 Computer Science Group Naval Postgraduate School Monterey, California 93940	1
4. Professor S. Jauregui, Code 62JA (Thesis Advisor) Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	5
5. LT. Lyle V. Rich, Code 52RS (Second Reader) Department of Computer Science Naval Postgraduate School Monterey, California 93940	3
6. Mr. J. Lynch Burrough's ADO Federal and Special Systems Group P.O. BOX 517 Paoli, Pa. 19301	1
7. Mr Carl Benson Naval Electronics Systems Engineering Center P. O. Box 80337 San Diego, California 92138	1

8. Mr. J. Lopata 1
Burrough's Corporation
P. O. Box 517
Paoli, Pa. 19301

9. Lt. Jerry M. Haggerty 1
103 Moran Circle
Monterey, California 93940

10. Lt. John M. Hartling 1
376 A. Bergin Drive
Monterey, California 93940

11. Naval Electronics Systems Command 1
Code PME 107
Washington, D. C. 20360
Attn: CAPT W. Flowers

12. Naval Electronics Systems Command 1
Code PME 107
Washington, D. C. 20360
Attn: Mr. R. Matterazza

13. Naval Electronics Systems Command 1
Code PME 107
Washington, D. C. 20360
Attn: CAPT H. Leavitt

Thesis 168148
H1116 Haggerty
c.1 Emulation of the AN/
UYK-7 tactical data com-
puter in the Burrough's
D-machine.
23 MAR 78 24877
6 SEP 78 25394
8 FEB 81 26385
5 DEC 84 29982

Thesis 168148
H1116 Haggerty
c.1 Emulation of the AN/
UYK-7 tactical data com-
puter in the Burrough's
D-machine.

DUDLEY KNOX LIBRARY
3 2768 00353584 0